Scalable Inference in Latent Gaussian Process Models

DISSERTATION

zur Erlangung des akademischen Grades

doctor rerum naturalium
(Dr. rer. nat.)
im Fach Informatik

eingereicht an der
Mathematisch-Naturwissenschaftlichen Fakultät
der Humboldt-Universität zu Berlin

von

M.Sc. Florian Wenzel

Präsidentin der Humboldt-Universität zu Berlin:
Prof. Dr.-Ing. Dr. Sabine Kunst

Dekan der Mathematisch-Naturwissenschaftlichen Fakultät:
Prof. Dr. Elmar Kulke

Gutachter/innen:
1. Prof. Dr. Marius Kloft
2. Prof. Dr. Manfred Opper
3. Prof. Dr. Stephan Mandt

Tag der mündlichen Prüfung: 25.11.2019

Abstract

Latent Gaussian process (GP) models help scientists to uncover hidden structure in data, express domain knowledge and form predictions about the future. These models have been successfully applied in many domains including robotics, geology, genetics and medicine. A GP defines a distribution over functions and can be used as a flexible building block to develop expressive probabilistic models. The main computational challenge of these models is to make inference about the unobserved latent random variables, that is, computing the posterior distribution given the data. Currently, most interesting Gaussian process models have limited applicability to big data. Naive inference typically scales cubically in the number of data points and exact computation of posterior is intractable and requires approximations like variational inference. Recent work around so-called sparse Gaussian process techniques has enabled the application of several GP models to big datasets. However, these methods are often unstable and inefficient since they are based on numerical approximations of the gradients of the variational objective.

This thesis develops a new efficient inference approach for latent GP models. Our new inference framework, which we call *augmented variational inference*, is based on the idea of considering an augmented version of the intractable GP model that renders the model conditionally conjugate. We show that inference in the augmented model is more efficient and, unlike in previous approaches, all updates can be computed in closed form.

The ideas around our inference framework facilitate novel latent GP models that lead to new results in language modeling, genetic association studies and uncertainty quantification in classification tasks. We develop new inference methods for binary *GP classification*, *multi-class GP classification* and the *Bayesian support vector machine*, which are up to two orders of magnitude faster than the state of the art. Furthermore, we propose two novel latent GP models. The *sparse GP linear mixed model* concerns feature selection in confounded data with binary labels and the *generalized dynamic topic model* can be used to analyze massive collections of text data by exploring topics that evolve over time.

Zusammenfassung

Latente Gauß-Prozess-Modelle (*latent Gaussian process models*) werden von Wissenschaftlern benutzt, um verborgenen Muster in Daten zu erkennen, Expertenwissen in probabilistische Modelle einfließen zu lassen und um Vorhersagen über die Zukunft zu treffen. Diese Modelle wurden erfolgreich in vielen Gebieten wie Robotik, Geologie, Genetik und Medizin angewendet. Gauß-Prozesse definieren Verteilungen über Funktionen und können als flexible Bausteine verwendet werden, um aussagekräftige probabilistische Modelle zu entwickeln. Dabei ist die größte Herausforderung, eine geeignete Inferenzmethode zu implementieren. Inferenz in probabilistischen Modellen bedeutet die A-Posteriori-Verteilung der latenten Variablen, gegeben der Daten, zu berechnen. Die meisten interessanten latenten Gauß-Prozess-Modelle haben zurzeit nur begrenzte Anwendungsmöglichkeiten auf großen Datensätzen. Naive Inferenzmethoden skalieren kubisch in der Anzahl der Datenpunkte und die exakte Berechnung der A-Posteriori-Verteilung ist nicht möglich und bedarf approximativer Methoden wie *variational inference*. Neue wissenschaftliche Veröffentlichungen um sogenannte *sparse Gaussian processes* haben es ermöglicht, dass einige Gauß-Prozess-Modelle auf großen Datenmengen angewendet werden können. Trotzdem sind diese Methoden oft instabil und ineffizient, da sie auf numerischen Approximationen des Gradienten der variationellen Zielfunktion beruhen.

In dieser Doktorarbeit stellen wir eine neue effiziente Inferenzmethode für latente Gauß-Prozess-Modelle vor. Unser neuer Ansatz, den wir *augmented variational inference* nennen, basiert auf der Idee, eine erweiterte (*augmented*) Version des Gauß-Prozess-Modells zu betrachten, welche bedingt konjugiert (*conditionally conjugate*) ist. Wir zeigen, dass Inferenz in dem erweiterten Modell effektiver ist und dass alle Schritte des *variational inference* Algorithmus in geschlossener Form berechnet werden können, was mit früheren Ansätzen nicht möglich war.

Unser neues Inferenzkonzept ermöglicht es, neue latente Gauß-Prozess-Modelle zu studieren, die zu innovativen Ergebnissen im Bereich der Sprachmodellierung, genetischen Assoziationsstudien und Quantifizierung der

Unsicherheit in Klassifikationsproblemen führen. In der vorliegenden Arbeit entwickeln wir neue Inferenzmethoden für *binary Gaussian process classification*, *multi-class GP classification* und die *Bayesian support vector machine*. Unsere Methode ist bis zu zwei Größenordnungen schneller als vorherige Methoden. Darüber hinaus entwickeln wir zwei neue latente Gauß-Prozess-Modelle. Das *sparse Gaussian process linear mixed model* kann für die Merkmalserkennung (*feature selection*) in Daten mit binären Labels verwendet werden und kann mit *confounding* Effekten umgehen. Mit dem *generalized dynamic topic model* können große Textdatensätze analysiert werden und Themen im Text gefunden werden, die sich über die Zeit verändern.

Acknowledgements

First of all, I would like to thank my PhD advisor, Prof. Marius Kloft, for his advice and encouragement. Marius gave me much freedom in following my research interests and supported me in every way to travel and collaborate with other researchers.

I am very grateful to Prof. Stephan Mandt for kindly inviting me to visit his group at Disney Research in Pittsburgh. We continued to collaborate on many projects and I always greatly enjoyed our countless stimulating discussions.

I want to express my sincere gratitude to Prof. Manfred Opper. I spend one and a half year as a visiting researcher in his lab. I was always intrigued by his wisdom and his great knowledge of old papers and "Physicists tricks" that often lead to new fascinating ideas.

My special thanks go to Théo Galy-Fajou with whom I shared an office for two years. We worked on many projects together and I always enjoyed Théo's enthusiasm for discussing new ideas and creating neat visualizations (especially animations) of our results.

I am grateful to the Machine Learning Group at Humboldt-Universität zu Berlin, the Machine Learning Group at TU Kaiserslautern and the Group of Artificial Intelligence at TU Berlin—especially Andreas, Burak, Christian, Cordula, Dimitra, Heike, Ludovica, Matthäus, Patrick, Robert for many engaging discussions. I also want to thank Nadia Sakhi and Dr. Robert Bamler for their valuable feedback on the manuscript.

Most of all, I would like to thank Ronja and my parents for supporting me in every conceivable way.

Finally, I acknowledge financial support of the German Bundesministerium für Bildung und Forschung (BMBF) under the awards 01IS18051A and 031B0187B, and the German Research Foundation (DFG) under the award KL 2698/2-1.

Contents

Basic notation

Unbolded x represents a single number, boldface $\boldsymbol{x}$ represents a vector, and capital unbolded X represents a matrix. An individual element of a vector is denoted with a subscript and without boldface. For example, the i-th element of a vector $\boldsymbol{x}$ is x_i. A bold lower-case letter with an index such as $\boldsymbol{x}_j$ represents a particular row of a matrix X (e.g. a data point $\boldsymbol{x}_j$ of the data matrix X).

Symbol	Description
f	A function $f : \mathbb{R}^d \to \mathbb{R}$.
$\boldsymbol{f}$	A vector of function values, whose i-th element is given by $f(\boldsymbol{x}_i)$, where $\boldsymbol{x}_i$ is the i-th data point.
$Y_{i\cdot}$	The i-th row of a matrix Y.
$Y_{\cdot j}$	The j-th column of a matrix Y.
K_{nn}	Shorthand for the kernel matrix of the training points.
K_{nm}	Shorthand for the kernel matrix between the training and inducing points.
K_{mm}	Shorthand for the kernel matrix between the inducing points.
$\mathrm{KL}(\,.\,\|\|\,.\,)$	Kullback-Leibler divergence.
$\mathcal{L}$	Evidence lower bound (ELBO).
$\mathcal{B}(.\|p)$	Bernoulli distribution with parameter p.
$\mathrm{Cat}(.\|p)$	Categorical distribution with parameter p.
$\mathrm{Dir}(.\|\alpha)$	Dirichlet distribution with concentration parameter α.
$\mathrm{Ga}(.\|a, b)$	Gamma distribution with shape parameter a and rate parameter b.
$\mathrm{GIG}(.\|a, b, c)$	Generalized inverse Gaussian distribution with parameters a, b, c.
$\mathcal{N}(.\|\mu, \Sigma)$	Gaussian distribution with mean parameter μ and covariance matrix Σ.
$\mathrm{PG}(.\|a, b, c)$	Pólya-Gamma distribution with parameters a, b, c.
$\mathrm{Po}(.\|\lambda)$	Poison distribution with rate parameter λ.
$\mathcal{O}(.)$	The big-O asymptotic complexity of an algorithm.

Part I

Overview

Chapter 1

Introduction

Gaussian processes (GPs) (Rasmussen and Williams, 2006) provide a popular Bayesian non-parametric framework, which can be used to address challenging machine learning problems. Because of their ability of accurately adapting to data and thus achieving high prediction accuracy while providing well calibrated uncertainty estimates, GPs are a standard method in several application areas including robotics (Sæmundsson, Hofmann, and Deisenroth, 2018; Beckers, Kulić, and Hirche, 2019), facial behavior analysis (Eleftheriadis et al., 2017), electrical engineering (Shepero et al., 2018; Pandit and Infield, 2018) and geospatial predictive modeling where they are known as *kriging* (Stein, 2012; Park and Apley, 2018).

Although deep learning has attracted tremendous attention from researchers in various fields such as language processing and computer vision, standard deep learning systems lack the ability to represent uncertainty in a mathematically sound way (Ghahramani, 2015). This is a critical shortcoming since in many applications areas including the life sciences as discussed by Herzog and Ostwald, 2013; Nuzzo, 2014, in biology and physics (Krzywinski and Altman, 2013), and self-driving cars (Michelmore, Kwiatkowska, and Gal, 2018), uncertainty information is crucial. In all these fields, we need systems that *"know when they don't know"*.

Gaussian processes, on the other hand, provide a mathematically sound approach to uncertainty representation. GPs are useful for representing random latent functions in probabilistic models. Computing the posterior distribution, i.e. the distribution of the latent function given the observed data, leads to sensible uncertainty estimates in the predictions for new data points (Rasmussen and Williams, 2006).

Because of providing well calibrated uncertainty estimates, GPs inspired and are widely used in many modern Bayesian deep learning approaches including deep kernel learning (Wilson et al., 2016), deep Gaussian processes (Damianou and Lawrence, 2013) and in the analysis of Bayesian neural networks (G. Matthews et al., 2018;

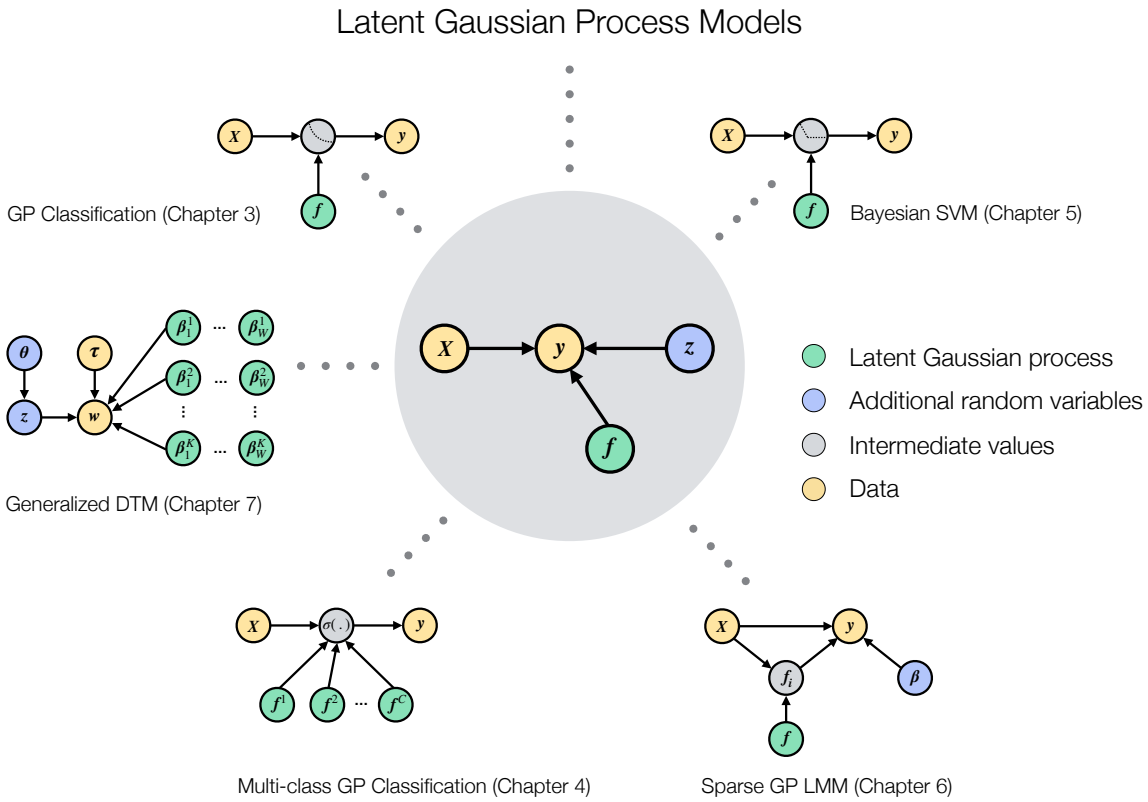

Figure 1.1: Latent Gaussian process models are probabilistic models that include one ore multiple latent GPs (green circles) and optionally additional latent random variables (blue circles), which are connected to the data (orange circles) via a possibly non-conjugate likelihood. In this thesis, we consider latent GP models of the structure as displayed in the center of the figure and develop an efficient inference approach for these models. Furthermore, we study five different models that are successfully applied in domains including binary and multi-class classification, genetic association studies and language modeling. All models can be cast as a latent GP model of the displayed structure and are amenable for our novel efficient inference approach.

Jacot, Gabriel, and Hongler, 2018). Moreover, GPs influenced other new interesting approaches to probabilistic function approximation in deep learning systems including neural processes (Garnelo et al., 2018) and variational implicit processes (Ma, Li, and Hernandez-Lobato, 2019).

But what is a Gaussian process? A GP defines a distribution over functions and is characterized by the property that any finite set of function values $f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)$ has a joint multivariate Gaussian distribution (Rasmussen and Williams, 2006). A GP is fully specified by its mean and covariance function, also called the *kernel function*. There are many possible choices of kernel functions, which give rise to a wide range of

different models. For instance, a GP based on the so-called *squared exponential kernel* function leads to a distribution over smooth functions, and using a *linear kernel* leads to a distribution over linear functions. GPs are well suited for developing interesting models. They can be utilized as a flexible building block in probabilistic modeling and become more expressive as the number of training examples increases. Through the choice of the kernel function, we can incorporate prior knowledge into the model and express a wide range of modeling assumptions.

For example, let us consider the problem of supervised binary classification. In this setting, the task is to predict the class label y associated to a new data point $\boldsymbol{x}$ given some observed data $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$. This task can be modeled by assuming a non-linear *latent function* $f(x)$, defined on the space of data points, that is connected to a corresponding class label y by a suitable *likelihood function* $p(y|f, x)$. We obtain a latent GP model by imposing a GP prior distribution over the latent function $p(f)$.

By connecting latent Gaussian processes with suitable likelihood functions, we can build a wide range of different models. In this thesis, we study several novel latent Gaussian process models, which are successfully applied in domains including binary and multi-class classification, genetic association studies and language modeling. As displayed in Fig.1.1, all models discussed in this thesis can be framed as latent GP models.

To predict new data, we have to make inference about the latent GP(s). This is a challenging problem and the main concern of this thesis. Recent trends in data availability in the sciences and technology have made it necessary to develop algorithms capable of processing massive data (John Walker, 2014). Despite the above mentioned advantages, currently, GP models have limited applicability to big data. Naive inference typically scales cubically in the number of data points, and exact computation of posterior and marginal likelihood is intractable.

Nevertheless, the combination of so-called sparse Gaussian process techniques with approximate inference methods, such as expectation propagation (EP) or the variational approach, have enabled certain GP models, e.g. GP classification, to datasets containing millions of data points (Hensman and Matthews, 2015; Dezfouli and Bonilla, 2015; Hernández-Lobato and Hernández-Lobato, 2016; Salimbeni, Eleftheriadis, and Hensman, 2018).

While these results are already impressive, we will show in this thesis that a speed-up of up to two orders of magnitude can be achieved. We develop a novel scalable

variational inference approach for latent GP models, which builds on an auxiliary variable augmentation of the model. An auxiliary variable is an additional latent variable that is added to a model such that the original model is recovered when this variable is integrated out. In our approach, the main idea is to consider an augmented version of the GP model that renders the model conditionally conjugate. In a conditionally conjugate model, all complete conditional distributions (the posterior distribution of one random variable given all the others), can be computed in closed form. Moreover, we show that inference in the augmented conditionally conjugate model is much easier than in the original model and demonstrate superior performance over the state of the art.

1.1 Organization and contributions

The main contribution of this thesis is to develop a new efficient variational inference approach based on auxiliary variable augmentation. This thesis also contributes novel latent GP models that lead to new results in language modeling, genetic association studies and uncertainty quantification in classification tasks.

Part I: Introduction and Overview

I start my dissertation in Chapter 1 with a short introduction to latent GP models. I lay out the basic inference problem and present the concept of sparse GP approximations.

In Chapter 2, I present a unifying view on a new efficient inference approach for latent GP models. The approach is called *augmented variational inference* and builds on an auxiliary variable augmentation technique. The main idea is to consider an augmented version of the latent GP model that renders the model conditionally conjugate. In the augmented model, we can derive an efficient variational inference algorithm, which is based on closed-form block coordinate ascent updates. These updates can be interpreted as natural gradient updates and are more efficient than ordinary Euclidean gradient updates. Additionally, the conditionally conjugate form of the model directly leads to an *exact Gibbs sampling* scheme.

Part II: Scalable Inference in Latent Gaussian Process Models

In the second part of my dissertation, I present five latent Gaussian process models. For each model, I develop a scalable inference method based on the principles laid out in the first part of the thesis and demonstrate superior performance over the state of

the art. Each chapter covers one model and can be read independently from the other chapters. For the sake of readability without interruptions, I briefly repeat the main concepts in each chapter while the details for the common ideas behind the *augmented variational inference* approach can be found in Chapter 2. The presented models and inference procedures are based on previous publications, which are indicated at the beginning of each chapter.

Chapter 3 considers Gaussian process classification based on the logistic likelihood function. I develop a stochastic variational inference method based on a Pólya-Gamma augmentation of the likelihood. Unlike in previous work, all updates are given in closed form and do not rely on numerical quadrature methods or sampling. In the experiments, I demonstrate that the *augmented variational inference* approach drastically improves computation time by up to two orders of magnitude while being competitive in terms of prediction performance.

Chapter 4 concerns multi-class GP classification. The multi-class classification problem is more complicated than the binary classification setting discussed in the previous chapter because it involves not only one latent GP, but one GP for each class. I introduce a novel multi-class GP classification model building on a modification of the softmax likelihood function. The new likelihood function has two advantages. First, it allows for an efficient auxiliary variable augmentation approach that renders the model conditionally conjugate. Second, it solves the calibration issue of previous work on scalable multi-class GP classification as it leads to better uncertainty quantification.

In Chapter 5, I develop a scalable inference method for a Bayesian version of the support vector machine (SVM). Advantages of the Bayesian approach over the common standard version of the SVM include automatic treatment of hyperparameters and direct quantification of the uncertainty of the prediction, which can be of great importance in practice. The inference method is based on a generalized inverse Gaussian augmentation and the experiments demonstrate superior performance over competing methods in terms of uncertainty quantification and speed.

Chapter 6 concerns feature selection in confounded data. This is an important problem in genetic association studies. The goal of this field is to find causal associations between high-dimensional vectors of genotypes, such as single nucleotide polymorphisms, and observable outcomes (feature selection). Genetic associations can be spurious, unreliable, and unreproducible when the data are subject to spurious correlations due to confounding. I propose the *sparse Gaussian process linear mixed model*, which finds relevant features while correcting for confounding. I discuss

two inference methods for two versions of the model. The first inference method is based on expectation propagation and is limited to small datasets. The second inference method builds on an *augmented variational inference* approach and scales to big datasets, but introduces a slight additional approximation error. The scalable *augmented variational inference* method was designed by me and implemented with the help of Lorenz Vaitl. In an experimental study on genetic data, I show that my new approach beats several baselines in terms of prediction performance, feature selection stability and finds features that are less correlated with the confounder.

In Chapter 7, I develop a new time dynamic topic model (DTM), which can be used for analyzing text data. DTMs model the evolution of prevalent themes in literature, online media and other forms of text over time. In the new *generalized dynamic topic model*, the time dynamics are governed by GPs. This allows for exploring topics that develop smoothly over time, that have a long-term memory or are temporally concentrated (for event detection). I discuss how to perform scalable approximate inference in this model. The experiments on several large-scale datasets show that my generalized model allows to find interesting patterns that were not accessible by previous approaches.

In Chapter 8, I conclude and discuss future work.

1.2 Brief introduction to Gaussian process models

In this section, we introduce the problem of learning from data via latent GP models. A GP defines a distribution over functions $f \sim \mathrm{GP}(\mu, k)$, where μ is the mean function and k is the kernel function. A GP is characterized by the property that any finite set of function values $f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)$ has a joint multivariate Gaussian distribution (Rasmussen and Williams, 2006). The distribution is completely specified by the mean function

$$\mathbb{E}[f(\boldsymbol{x})] = \mu(\boldsymbol{x})$$

and the kernel function (also called covariance function)

$$\mathrm{Cov}[f(\boldsymbol{x}), f(\boldsymbol{x}')] = k(\boldsymbol{x}, \boldsymbol{x}').$$

In practice, we usually assume that the mean function is simply zero since uncertainty about the mean function can be taken into account by adding an extra term to the kernel (Rasmussen and Williams, 2006).

Each kernel function corresponds to a different set of assumptions we make about the function we wish to model (Rasmussen and Williams, 2006; Duvenaud, 2014). In particular, the type of kernel function determines the type of latent functions we expect to see and determines how the model generalizes to new data. There are many possible choices of kernel functions. For instance, the *squared exponential kernel* is defined by

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp\left(-\frac{(\boldsymbol{x} - \boldsymbol{x}')^2}{2\ell^2}\right),$$

where the variance parameter σ_f^2 and the length scale parameter ℓ are the *hyperparameters* of the kernel. Using this kernel leads to a prior distribution over smooth functions. The *linear kernel*

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 (\boldsymbol{x} - c)^\top (\boldsymbol{x}' - c),$$

with hyperparameters σ_f^2 and c leads to a distribution over linear functions. In Fig. 1.2, left, we show several functions drawn from a GP prior using a squared exponential kernel with variance and length scale parameters set to one.

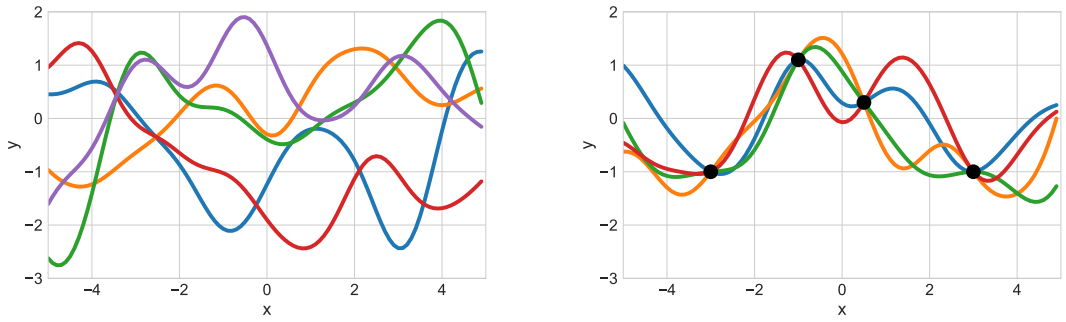

Figure 1.2: LEFT: Functions drawn from a *Gaussian process prior* using a squared exponential kernel. RIGHT: Functions drawn from the *posterior* of a GP regression model with no observation noise, after conditioning on four observations (black dots).

Latent Gaussian process models. In this thesis, we are interested in probabilistic models that consist of a latent function f that is defined on the data points $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^\top \in \mathbb{R}^{n \times d}$. We assume a prior GP distribution on the latent function $f \sim \mathrm{GP}(0, k)$ and the data labels $\boldsymbol{y} = (y_1, \ldots, y_n)$ are connected to f via a *factorizing*

likelihood $p(\boldsymbol{y}|f)$. The generative model is

$$\begin{aligned} f &\sim \mathrm{GP}(0, k) \\ \boldsymbol{y} &\sim p(\boldsymbol{y}|f, X) = \prod_{i=1}^{n} p(y_i|f(\boldsymbol{x}_i)). \end{aligned} \tag{1.1}$$

In this thesis, we refer to this class of models as *latent Gaussian process models*.

A wide range of models can be obtained by choosing different likelihood functions. For instance using a Gaussian likelihood

$$p(y_i|f(\boldsymbol{x}_i)) = \mathcal{N}(y_i|f(\boldsymbol{x}_i), \sigma^2), \tag{1.2}$$

where $y_i \in \mathbb{R}$ and σ^2 is an additional parameter, leads to the standard (Gaussian likelihood) GP regression model. A binary classification model can be obtained by using a Bernoulli likelihood

$$p(y_i|f(\boldsymbol{x}_i)) = \mathcal{B}(y_i|\sigma(y_i f(\boldsymbol{x}_i))),$$

where $y_i \in \{-1, 1\}$ and $\sigma(.)$ is an activation function mapping the GP outputs to probabilities in the interval $[0, 1]$. More complicated likelihoods can be obtained by considering models with additional latent variables $\boldsymbol{z} = (z_1, \ldots, z_k)$ (e.g. hierarchical models). For instance, in Chapter 7 we propose a language model that has a hierarchical structure based on a set of multiple latent GPs and additional latent variables, which model certain aspects of a text document (see Fig. 7.1).

For the sake of simplicity, in the following of the introduction, we only consider latent GP models of the form given in Eq. 1.1 and omit potential additional latent variables and assume a single latent GP f. Note that the ideas developed in the first two chapters also apply to models with multiple latent GPs and models with additional latent variables[1].

In the following, we use the shorthand $f_i = f(x_i)$ and omit that we are always conditioning on the data points X.

Inference in latent GP models. The main goal is to compute the *posterior distribution* of the latent GP, which, in principle, is given by Bayes' rule

$$p(\boldsymbol{f}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f})}{p(\boldsymbol{y})}. \tag{1.3}$$

[1]In particular, models with additional latent variables can still be viewed from the perspective of having a single (complicated) likelihood in the form of Eq. 1.1 by considering the marginal likelihood $p(\boldsymbol{y}|f, X) = \int p(\boldsymbol{y}|\boldsymbol{z}, f, X)p(\boldsymbol{z}|f, X)\mathrm{d}\boldsymbol{z}$.

We can make predictions about unseen data $X_* = (\boldsymbol{x}_{*1}, \ldots, \boldsymbol{x}_{*k})^\top$ by first computing the distribution of the latent function values corresponding to the test cases

$$p(\boldsymbol{f}_*|\boldsymbol{y}, X_*) = \int p(\boldsymbol{f}_*|\boldsymbol{f}, X_*)p(\boldsymbol{f}|\boldsymbol{y})\mathrm{d}\boldsymbol{f}$$

and then using this distribution over $\boldsymbol{f}_*$ to compute the *predictive distribution*

$$p(\boldsymbol{y}_*|\boldsymbol{y}, X_*) = \int p(\boldsymbol{y}_*|\boldsymbol{f}_*)p(\boldsymbol{f}_*|\boldsymbol{y}, X_*)\mathrm{d}\boldsymbol{f}_*.$$

In the case of Gaussian GP regression, i.e. using the likelihood given in (1.2), the posterior and predictive distribution can be computed in closed form (Rasmussen and Williams, 2006). In this case, the likelihood is *conjugate* to the GP prior, that means that the posterior distribution is in the same family as the prior distribution. Hence, in the context of GP regression, the posterior of the function values $\boldsymbol{f}$ is again Gaussian distributed. Moreover, the predictive distribution is given by

$$p(\boldsymbol{y}_*|\boldsymbol{y}, X_*) = \mathcal{N}(\boldsymbol{\mu}_*, \Sigma_*) \tag{1.4}$$

with

$$\begin{aligned}
\boldsymbol{\mu}_* &= K\left(X_*, X\right)\left(K(X, X) + \sigma^2 I\right)^{-1}\boldsymbol{y} \\
\Sigma_* &= K\left(X_*, X_*\right) - K\left(X_*, X\right)\left(K(X, X) + \sigma_n^2 I\right)^{-1} K\left(X, X_*\right),
\end{aligned}$$

where $K(X, X')$ denotes the *kernel matrix* resulting from evaluating the kernel function between points $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)^\top$ and $X' = (\boldsymbol{x}'_1, \ldots, \boldsymbol{x}'_{N'})^\top$, which is given by $K(X, X')_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}'_j)$. In Fig. 1.2, right, we show functions drawn from the posterior predictive distribution (1.4) upon observing four data points in a GP regression model using a squared exponential kernel.

For other *non-conjugate* likelihoods, computing the posterior (1.3) is often intractable and requires *approximate inference*. Many different approaches to approximate inference in GP models have been proposed including sampling based methods, e.g. elliptic slice sampling (Murray, Adams, and MacKay, 2010) and sampling using control variates (Lawrence, Rattray, and Titsias, 2009), as well as variational methods, e.g. based on expectation propagation (Hernández-Lobato and Hernández-Lobato, 2016) or variational inference (Hensman, Fusi, and Lawrence, 2013; Salimbeni, Eleftheriadis, and Hensman, 2018).

Another issue is that, even in the simple GP regression model, inference is slow and does not scale to big datasets. Due to the inversion of the kernel matrix, naive

inference in latent GP models typically has computational complexity $\mathcal{O}(n^3)$, where n is the number of data points.

In this thesis, we focus on variational inference and discuss how these problems can be circumvented.

Variational inference. We employ *variational inference* to deal with the intractable posterior (1.3). Variational inference maps the infeasible inference problem to a feasible optimization problem (see e.g. Blei, Kucukelbir, and McAuliffe, 2017). The idea is to choose a family of tractable variational distributions $q_\theta(\boldsymbol{f})$, where $\boldsymbol{\theta}$ denote the variational parameters of the distribution family, and then select the best approximation of the true posterior $p(\boldsymbol{f}|\boldsymbol{y})$ by minimizing the Kullback-Leibler divergence between the variational distribution and the posterior

$$\mathrm{KL}(q_\theta(\boldsymbol{f})||p(\boldsymbol{f}|\boldsymbol{y})) = \mathbb{E}_{q_\theta(\boldsymbol{f})}[\log q_\theta(\boldsymbol{f}) - \log p(\boldsymbol{f}|\boldsymbol{y})]$$

with respect to the variational parameters $\boldsymbol{\theta}$. This is equivalent to maximizing a lower bound on the log marginal likelihood, known as evidence lower bound (ELBO)

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{q_\theta(\boldsymbol{f})}[\log p(\boldsymbol{y}, \boldsymbol{f}) - \log q_\theta(\boldsymbol{f})] \leq \log p(\boldsymbol{y}).$$

Sparse variational approximation. *Sparse Gaussian processes* address the issue of the cubic computational complexity of inference in GP models. Different approaches to sparse approximations of GPs have been discussed in literature including the early work by Csató and Opper, 2002 and Csató, 2002, and Seeger, Williams, and Lawrence, 2003. Based on that work, we employ a variational approach and follow the line of research by Snelson and Ghahramani, 2005; Titsias, 2009; Hensman, Fusi, and Lawrence, 2013. We replace the latent GP f by a sparse approximation building on a set of inducing points. This reduces the complexity to $\mathcal{O}(m^3)$, where m is the number of inducing points.

The sparse GP approximation is obtained as follows. We start with the latent GP model (1.1) and introduce an inducing-point augmentation to the latent GP f. The new random variable consists of m additional input-output pairs $(\boldsymbol{z}_1, u_1), \ldots, (\boldsymbol{z}_m, u_m)$, termed as *inducing inputs* and *inducing variables*. The function values of the GP $\boldsymbol{f}$ and the inducing variables $\boldsymbol{u} = (u_1, \ldots, u_m)$ are connected via

$$\begin{aligned} p(\boldsymbol{f}|\boldsymbol{u}) &= \mathcal{N}\left(\boldsymbol{f}|K_{nm}K_{mm}^{-1}\boldsymbol{u}, \widetilde{K}\right) \\ p(\boldsymbol{u}) &= \mathcal{N}\left(\boldsymbol{u}|0, K_{mm}\right), \end{aligned} \tag{1.5}$$

where K_{mm} is the kernel matrix resulting from evaluating the kernel function between all inducing inputs, K_{nm} is the cross-kernel matrix between inducing inputs and training points, and $\widetilde{K} = K_{nn} - K_{nm}K_{mm}^{-1}K_{mn}$. Including the inducing points in our model gives the augmented joint distribution

$$p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u}).$$

Note that the original model (1.1) can be recovered by marginalizing $\boldsymbol{u}$.

We aim to approximate the posterior of the sparse GP $p(\boldsymbol{u}|\boldsymbol{y})$ and apply the methodology of variational inference to the augmented joint distribution $p(\boldsymbol{y}, \boldsymbol{u}, \boldsymbol{f})$. We assume the variational distribution family $q(\boldsymbol{u}, \boldsymbol{f}) = q(\boldsymbol{u})p(\boldsymbol{f}|\boldsymbol{u})$, where $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{m}, S)$ is a variational Gaussian distribution with variational mean parameter $\boldsymbol{m}$ and variational covariance parameter S. Note that the factor $p(\boldsymbol{f}|\boldsymbol{u})$ is fixed and given by Eq. 1.5.

The final variational lower bound on the evidence (ELBO) is given by

$$\log p(y) \geq \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})}[\log p(\boldsymbol{y}|\boldsymbol{f})] - \mathrm{KL}\left(q(\boldsymbol{u})||p(\boldsymbol{u})\right). \tag{1.6}$$

The inducing point locations Z are additional hyperparameters and are either fixed or optimized via maximizing the fitted variational lower bound as a function of the inducing point locations (see Section 2.2).

Chapter 2

Augmented Variational Inference for Gaussian Process Models

In this chapter, we propose an *unified framework* for efficient inference in latent GP models. Inference in latent GP models that go beyond simple regression is challenging due to the non-conjugate likelihood. Previous scalable inference methods (e.g. Hensman and Matthews, 2015; Dezfouli and Bonilla, 2015; Hernández-Lobato and Hernández-Lobato, 2016; Salimbeni, Eleftheriadis, and Hensman, 2018) typically build on approximating the likelihood by sampling or numerical quadrature, thus, preventing efficient optimization.

We develop a different approach that aims on translating the intractable non-conjugate model into an easier conditionally conjugate model by adding auxiliary random variables to the model. An auxiliary variable is an additional latent variable that is added to a model such that the original model is recovered when this variable is marginalized out. We aim to find an augmentation that renders the model conditionally conjugate. Inference in the augmented conditionally conjugate model is much easier than in the original model. We develop an efficient variational inference method based on block coordinate updates, which can be computed in closed form. The updates correspond to natural gradient updates, which are more efficient than ordinary Euclidean gradient updates, which are used in previous approaches.

In the following of this chapter, we introduce the main ideas behind our new *augmented variational inference* approach. This chapter is meant to provide a high-level motivation of our approach and focuses on the general concepts. In part II of the thesis, we show that this framework is actually useful in practice. We study five different latent GP models and develop scalable inference methods based on the principles laid out in this chapter.

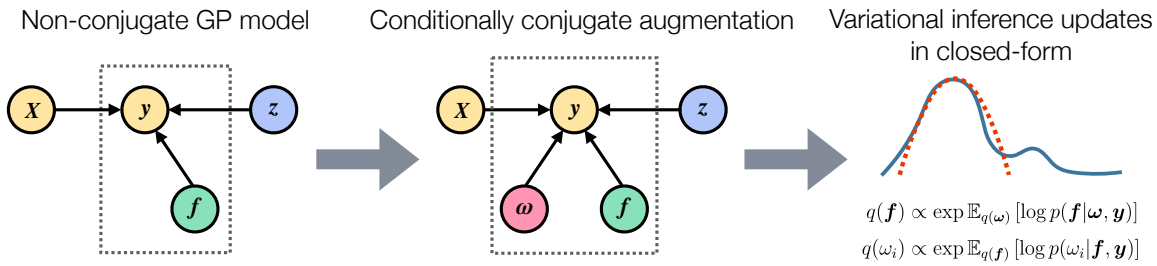

Figure 2.1: How can we perform scalable inference in latent Gaussian process models? We propose an *augmented variational inference* approach, which leads to efficient variational inference updates given in closed form. We start with the non-conjugate latent GP model (LEFT), where the latent GP is denoted by f, (optional) additional random variables are denoted by z and the data is (X, y). In the first step, we add auxiliary random variables ω to the model that render the model conditionally conjugate (MIDDLE). In the conditionally conjugate model, all complete conditional distributions (the posterior distribution of one random variable given all the others) are tractable. In the last step (RIGHT), we obtain an approximation of the posterior. We leverage conditional conjugacy to derive efficient variational inference updates, which are given in closed form.

2.1 A conditionally conjugate formulation via auxiliary variable augmentation

The main idea of the *augmented variational inference* approach is to add auxiliary random variables to the latent GP model such that inference becomes easier. In particular, we aim to find an augmented representation of the model that renders the model conditionally conjugate. Let $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_n)$ be a set of auxiliary random variables, which augment the original model (1.1) leading to a joint distribution of the form

$$p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{\omega}) = \prod_i p(y_i|f_i, \omega_i)p(\omega_i)p(\boldsymbol{f}). \tag{2.1}$$

The original model can be restored by marginalizing $\boldsymbol{\omega}$, i.e. $p(\boldsymbol{y}, \boldsymbol{f}) = \int p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega}$.

In our work, the goal is to find an augmentation $\boldsymbol{\omega}$ such that the augmented likelihood $p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\omega})$ becomes conjugate to the prior distributions $p(\boldsymbol{f})$ and $p(\boldsymbol{\omega})$, i.e. the complete conditional distributions $p(\boldsymbol{f}|\boldsymbol{\omega}, \boldsymbol{y})$ and $p(\boldsymbol{\omega}|\boldsymbol{f}, \boldsymbol{y})$ are in the same family as their associated priors. Furthermore, we seek for an augmentation the allows us to compute expectations of the log complete conditional distributions, given in Eq. 2.3, in closed form. Later we will see that these requirements are sufficient for obtaining efficient variational inference updates (see Eq. 2.3).

As a direct consequence from the above requirements, we seek for augmentations that render the likelihood to be a squared exponential function in f_i, i.e.

$$p(y_i|f_i,\omega_i) \propto \exp\left(a(y_i,\omega_i) + b(y_i,\omega_i)f_i + c(y_i,\omega_i)f_i^2\right), \tag{2.2}$$

where a and b are arbitrary factors that only depend on the label y_i and the auxiliary variable ω_i. An augmented likelihood of this form leads to a Gaussian complete conditional $p(\boldsymbol{f}|\boldsymbol{\omega},\boldsymbol{y})$. Additional to the conditional conjugacy in $\boldsymbol{f}$, we seek for an augmentation that also leads to a tractable complete conditional distribution in the augmented variables $p(\boldsymbol{\omega}|\boldsymbol{f},\boldsymbol{y})$.

To summarize, our method is based on finding a *suitable* augmentation of the original model. In this section, we have established the requirements that define a *suitable* augmentation. The goal is to find an augmentation $\boldsymbol{\omega}$, which

- renders the model conditionally conjugate and
- allows us to compute expectations of the complete conditionals (given in Eq. 2.3) in closed-from.

These goals serve as a guide in the process of finding suitable augmentations. In the next chapters, we will see that such augmentations can be indeed found for a variety of different models and lead to a substantial decrease in computation time when performing variational inference.

Note that a suitable augmentation can involve more than one set of auxiliary variables. For instance, in Chapter 4 we develop a hierarchical augmentation of a multi-class GP classification model, which involves three different sets of auxiliary variables. For the sake of simplicity, in this chapter, we focus on the case of having one set of auxiliary variables $\boldsymbol{\omega} = (\omega_1,\ldots,\omega_n)$ that factorizes the likelihood in the form of Eq. 2.1.

2.2 Variational inference in the augmented model

For now, let us assume we have found an augmentation $\boldsymbol{\omega}$ that fulfills the requirements defined in Section 2.1. We show that in this case, the posterior can be efficiently approximated using variational inference.

We assume a variational distribution where the latent GP $\boldsymbol{f}$ and the augmentation variables are decoupled, i.e. $q(\boldsymbol{f},\boldsymbol{\omega}) = q(\boldsymbol{f})q(\boldsymbol{\omega})$. Following standard results (Blei, Kucukelbir, and McAuliffe, 2017), the optimal variational distribution of $\boldsymbol{\omega}$ factorizes,

i.e. $q(\boldsymbol{\omega}) = \prod_i q(\omega_i)$ and the variational distributions can be iteratively optimized by the block-coordinate ascent updates given by

$$\begin{aligned} q(\boldsymbol{f}) &\propto \exp \mathbb{E}_{q(\boldsymbol{\omega})} \left[\log p(\boldsymbol{f}|\boldsymbol{\omega}, \boldsymbol{y})\right] \\ q(\omega_i) &\propto \exp \mathbb{E}_{q(\boldsymbol{f})} \left[\log p(\omega_i|\boldsymbol{f}, \boldsymbol{y})\right]. \end{aligned} \tag{2.3}$$

The assumptions from the previous section imply that these updates are computed in closed form. Furthermore, from the structure of the augmented likelihood (2.2), it follows that the variational distribution $q(\boldsymbol{f})$ is in the same family as the corresponding complete conditional and, thus, $q(\boldsymbol{f})$ is a variational Gaussian distribution.

In order to scale to big datasets we employ *stochastic variational inference* (Hoffman et al., 2013) and replace the original latent GP f by a sparse approximation u as introduced in Section 1.2. Hoffman et al., 2013 have shown in the setting, where the complete conditionals are in the exponential family, that the coordinate updates (2.3) can be directly interpreted as natural gradient updates with a learning rate of one. In stochastic variational inference, the variational objective is optimized via stochastic optimization based on stochastic natural gradients. In each iteration, we use mini-batches of the data and obtain a noisy version of the natural gradient. In this setting, learning rates slightly less than one have to be chosen.

Optimization of hyperparameters. The hyperparameters of our model include kernel hyperparameters and inducing point locations. The hyperparameters can be either fixed before training or optimized via a variational expectation maximization (EM) approach.

We select the optimal kernel hyperparameters by maximizing the marginal likelihood $p(\boldsymbol{y}|h)$, where h denotes the set of hyperparameters (this approach is called empirical Bayes (Maritz and Lwin, 1989)). We follow an approximate approach and optimize the fitted variational lower bound $\mathcal{L}(h)$ as a function of h by alternating between optimization steps w.r.t. the variational parameters and the hyperparameters (see e.g. Mandt, Hoffman, and Blei, 2016).

2.3 On the quality of the augmented approximation

The main advantage of the *augmented variational inference* approach is, that it leads to variational updates that are computed in closed form and can be interpreted as efficient block coordinate updates. Hence, our inference method is often faster and

more stable than competing methods that apply variational inference to the original (non-augmented) latent GP model.

The price we have to pay for faster inference is an additional approximation error compared to the variational inference solution of the original model. In this section, we discuss the additional augmentation gap of the ELBO introduced by the auxiliary variables $\boldsymbol{\omega}$. For the sake of simplicity, we consider here the non-sparse case, i.e. the inducing points equal the training points ($\boldsymbol{f} = \boldsymbol{u}$). However, it is straightforward to extend the results also to the sparse case.

In a direct application of variational inference to the latent GP model (Eq. 1.1), one tries to approximate the posterior directly by a variational Gaussian $q^*(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{\mu}^*, \Sigma*)$ without using an auxiliary variable augmentation. This leads to the optimization problem

$$\arg\min_{q(\boldsymbol{f})} \mathrm{KL}(q(\boldsymbol{f})||p(\boldsymbol{f}|\boldsymbol{y})) = \arg\max_{q(\boldsymbol{f})} \mathbb{E}_{q(\boldsymbol{f})}[\log p(\boldsymbol{y}, \boldsymbol{f}) - \log q(\boldsymbol{f})].$$

Optimizing the *original ELBO* $\mathcal{L}_{\mathrm{orig}}(q)$ (right-hand side of the equation above) is in general not tractable. For some models, e.g. GP classification models (Hensman and Matthews, 2015), the ELBO can be approximated by sampling or numerical quadrature. For the sake of the analysis, we now assume that we have an algorithm that allows us to optimize the original ELBO and we denote the optimizer with $q^*_{\mathrm{orig}}(\boldsymbol{f})$.

In our approach, we apply variational inference to the augmented model and optimize the *augmented ELBO*

$$\mathcal{L}_{\mathrm{augmented}}(q) = \mathbb{E}_{q(\boldsymbol{f})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{\omega}) - \log q(\boldsymbol{f}) - \log q(\boldsymbol{\omega})].$$

We look for the best distribution $q^*(\boldsymbol{f}, \boldsymbol{\omega}) = q^*(\boldsymbol{f})q^*(\boldsymbol{\omega})$ that factorizes in the augmented auxiliary variables $\boldsymbol{\omega}$ and the original function $\boldsymbol{f}$. This approach also yields a Gaussian approximation $q^*(\boldsymbol{f})$ as a factor in the optimal density. Of course $q^*(\boldsymbol{f})$ will be different from the "optimal" approximation $q^*_{\mathrm{orig}}(\boldsymbol{f})$ that maximizes the original ELBO.

It would be interesting to see how the maximized ELBOs of the two variational approaches, which both give a lower bound on the likelihood of the data, differ. Unfortunately, such a computation would require the knowledge of the optimal $q^*_{\mathrm{orig}}(\boldsymbol{f})$. However, we can obtain some estimate of this difference by directly comparing both

bounds using the *same* Gaussian density $q(\boldsymbol{f})$. In this case, we obtain

$$\begin{aligned}
&\mathcal{L}_{\text{orig}}(q(\boldsymbol{f})) - \mathcal{L}_{\text{augmented}}(q(\boldsymbol{f})q(\boldsymbol{\omega})) \\
&= \mathbb{E}_{q(\boldsymbol{f})}[\log p(\boldsymbol{y}, \boldsymbol{f}) - \log q(\boldsymbol{f})] - \mathbb{E}_{q(\boldsymbol{f})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{\omega}) - \log(q(\boldsymbol{f})q(\boldsymbol{\omega}))] \\
&= \mathbb{E}_{q(\boldsymbol{f})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{f}) - \log p(\boldsymbol{y}|\boldsymbol{f}, \boldsymbol{\omega}) - \log p(\boldsymbol{\omega}) + \log q(\boldsymbol{\omega})] \\
&= \mathbb{E}_{q(\boldsymbol{f})q(\boldsymbol{\omega})}[-\log p(\boldsymbol{\omega}|\boldsymbol{f}, \boldsymbol{y}) + \log q(\boldsymbol{\omega})] \\
&= \mathbb{E}_{q(\boldsymbol{f})}[\mathrm{KL}\left(q(\boldsymbol{\omega})||p(\boldsymbol{\omega}|\boldsymbol{f}, \boldsymbol{y})\right)].
\end{aligned}$$

The *augmentation gap* is small if, on average, the variational approximation $q(\boldsymbol{\omega})$ is close to the posterior $p(\boldsymbol{\omega}|\boldsymbol{f}, \boldsymbol{y})$.

We could, however, argue that asymptotically, in the limit of a large number of data, the predictions given by both densities may not differ a lot, as the posterior uncertainty for both densities should become small. Neglecting this uncertainty in the expectation of the log-likelihood in the Gaussian variational approximation shows that the approximation of the posterior mean in $q^*(\boldsymbol{f})$ becomes equal to the MAP estimator (Opper and Archambeau, 2009). If we neglect the posterior uncertainty (and the extra error caused by sparsity) in our approximation, we simply arrive at an *exact* EM algorithm for computing the same MAP estimator of the model. This is because in the E-step, we can compute the expectation over the augmentation variables $\boldsymbol{\omega}$ exactly. We might, however, get some difference in the covariance structure of the two Gaussian densities $q^*(\boldsymbol{f})$ and $q(\boldsymbol{f})$, which is not expected to cause a major deterioration of the predictions.

For the latent GP models discussed in the next chapters, we confirm empirically that the additional approximation error is small in practice and the predictive performance of our approach is competitive.

2.4 Gibbs sampling

Since our augmented model is conditionally conjugate we can directly derive a Gibbs sampling scheme. In order to sample from the exact posterior, we alternate between drawing a sample from each complete conditional distribution

$$\boldsymbol{f}_t \sim p(\boldsymbol{f}|\boldsymbol{\omega}_{t-1}, \boldsymbol{y})$$

and

$$\boldsymbol{\omega}_t \sim p(\boldsymbol{\omega}|\boldsymbol{f}_t, \boldsymbol{y}),$$

where $\boldsymbol{\omega}_0$ is appropriately initialized. The augmented variables are naturally marginalized out and asymptotically, the latent GP samples will be from the true posterior.

In this thesis, we focus on the methodology of variational inference since it can be scaled to big datasets more easily. However, the Gibbs sampling scheme is a useful tool to empirically assess the quality of our approximate posterior on small datasets.

Part II

Scalable Inference in Latent Gaussian Process Models

Chapter 3

Gaussian Process Classification

We begin our study of latent GP models with a GP classification model. We propose a scalable stochastic variational approach building on Pólya-Gamma data augmentation and inducing points. Unlike previous approaches, we obtain closed-form updates based on natural gradients, which lead to efficient optimization. We evaluate the algorithm on real-world datasets containing up to 11 million data points and demonstrate that it is up to two orders of magnitude faster than the state of the art while being competitive in terms of prediction performance. The code is available via Github[1].

This chapter is based on:

> F. Wenzel*, T. Galy-Fajou*, C. Donner, M. Kloft, and M. Opper (2019). "Efficient Gaussian Process Classification Using Pólya-gamma Data Augmentation". In: *AAAI Conference on Artificial Intelligence.*

We consider a binary GP Classification model, which is defined as follows. Let $X = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n)^\top \in \mathbb{R}^{n \times d}$ be the d-dimensional training points with labels $\boldsymbol{y} = (y_1, \dots, y_n) \in \{-1, 1\}^n$. The likelihood of the labels is

$$p(\boldsymbol{y}|\boldsymbol{f}) = \prod_{i=1}^{n} \sigma(y_i f(\boldsymbol{x}_i)), \tag{3.1}$$

where $f \sim \mathrm{GP}(0, k)$ is a latent GP function and $\sigma(z)$ is an activation function, which maps the latent function values to the interval $[0, 1]$. In our work, we consider the logistic function $\sigma(z) = (1 + \exp(-z))^{-1}$.

Currently, GP classification has limited applicability to big data. Naive inference typically scales cubically in the number of data points, and exact computation of the posterior and marginal likelihood is intractable.

[1] `https://github.com/theogf/AugmentedGaussianProcesses.jl`

*Equal contributions.

Nevertheless, the combination of so-called sparse Gaussian process techniques (cf. Section 1.2) with approximate inference methods, such as expectation propagation (EP) or the variational approach, have enabled GP classification for datasets containing millions of data points (Hernández-Lobato and Hernández-Lobato, 2016; Salimbeni, Eleftheriadis, and Hensman, 2018).

While these results are already impressive, we will show that our *augmented variational inference* approach leads to a speed-up of up to two orders of magnitude. We show that the logistic GP classification model can be augmented by Pólya-Gamma random variables (Polson, Scott, and Windle, 2013) leading to a conditionally conjugate model.

As outlined in Section 2.1, the augmentation approach gives directly rise to coordinate ascent updates and the optimization procedure can be interpreted as a natural-gradient approach to variational inference. Unlike previous approaches, the natural gradient updates can be computed in closed form leading to a fast and stable algorithm, which is simple to implement.

3.1 Background and related work

Gaussian process classification. Hensman and Matthews (2015) consider Gaussian process classification with a probit inverse link function and suggest a variational Gaussian model that builds on inducing points. By employing automatic differentiation, Salimbeni, Eleftheriadis, and Hensman (2018) generalize this approach to apply natural gradients in non-conjugate GP models. Khan and Nielsen (2018) consider natural gradient updates in the setting of variational inference with exponential families. Unlike our approach, these methods do not benefit from closed-form updates and have to resort to numerical approximations. Moreover, our approach has the advantage that a higher learning rate close to one can be chosen since our updates can be interpreted as block-coordinate ascent updates.

Izmailov, Novikov, and Kropotov, 2018 use tensor train decomposition to train GP models with billions of inducing points. The updates are not computed in closed form and they do not use natural gradients.

Dezfouli and Bonilla (2015) propose a general automated variational inference approach for sparse GP models with non-conjugate likelihoods. Since they follow a black box approach and do not exploit model specific properties, they do not employ efficient optimization techniques.

Hernández-Lobato and Hernández-Lobato (2016) follow an expectation propagation approach based on inducing points and have a similar computational cost as Hensman and Matthews (2015).

Pólya-Gamma data augmentation. Polson, Scott, and Windle (2013) introduced the idea of data augmentation in logistic models using the class of Pólya-Gamma distributions. This allows for exact inference via Gibbs sampling or approximate variational inference schemes (Scott and Sun, 2013).

Linderman, Johnson, and Adams (2015) extend this idea to multinomial models and discuss the application for Gaussian processes with multinomial observations but their approach does not scale to big datasets and they do not consider the concept of inducing points.

3.2 The augmentation

Due to the analytically inconvenient form of the likelihood function, inference for logistic GP classification is a challenging problem. We remedy this issue by considering an augmented representation of the original model. The augmented model is advantageous since it leads to efficient closed-form updates in our variational inference scheme.

Polson, Scott, and Windle (2013) introduced the class of Pólya-Gamma random variables and proposed a data augmentation strategy for inference via Gibbs sampling in models with binomial likelihoods. The augmented model has the appealing property that the likelihood of the latent function values $\boldsymbol{f}$ is proportional to a Gaussian density when conditioned on the augmented Pólya-Gamma variables. Furthermore, the auxiliary Pólya-Gamma variables conditioned on the GP values $\boldsymbol{f}$ are again Pólya-Gamma distributed. The Pólya-Gamma augmentation satisfies our requirements from Section 2.1 and hence can be utilized to derive an efficient approximate inference algorithm.

The Pólya-Gamma distribution is defined as follows. The random variable $\omega \sim \mathrm{PG}(b, 0)$, $b > 0$ is defined by the moment generating function

$$\mathbb{E}_{\mathrm{PG}(\omega|\, b,0)}[\exp(-\omega t)] = \frac{1}{\cosh^b(\sqrt{t/2})}. \tag{3.2}$$

It can be shown that this is the Laplace transform of an infinite convolution of gamma distributions. The definition is related to our problem by the fact that the logistic likelihood can be written in a form that involves the $cosh(.)$ function, namely $\sigma(z_i) =$

$\exp(\frac{1}{2}z_i)(2\cosh(\frac{z_i}{2}))^{-1}$. In the following we derive a representation of the logistic likelihood in terms of Pólya-Gamma variables.

First, we define the general $\mathrm{PG}(b, c)$ class that is derived by an exponential tilting of the $\mathrm{PG}(b, 0)$ density, and it is given by

$$\mathrm{PG}(\omega|\, b, c) \propto \exp(-\frac{c^2}{2}\omega)\mathrm{PG}(\omega|\, b, 0).$$

From the moment generating function (3.2), the first moment can be directly computed

$$\mathbb{E}_{PG(\omega|b,c)}[\omega] = \frac{b}{2c}\tanh\left(\frac{c}{2}\right).$$

For the subsequently presented variational algorithm, these properties suffice and the full representation of the Pólya-Gamma density $\mathrm{PG}(\omega|b, c)$ is not required.

We now adapt the data augmentation strategy based on Pólya-Gamma variables for the GP classification model. To do this, we write the non-conjugate logistic likelihood function (3.1) in terms of Pólya-Gamma variables

$$\begin{aligned}\sigma(z_i) = (1+\exp(-z_i))^{-1} &= \frac{\exp(\frac{1}{2}z_i)}{2\cosh(\frac{z_i}{2})} \\ &= \frac{1}{2}\int \exp\left(\frac{z_i}{2} - \frac{z_i^2}{2}\omega_i\right) p(\omega_i)\mathrm{d}\omega_i, \end{aligned} \tag{3.3}$$

where $p(\omega_i) = \mathrm{PG}(\omega_i|1, 0)$ and by making use of (3.2). For more details see Polson, Scott, and Windle (2013). Using this identity and substituting $z_i = y_i f(x_i)$ we augment the joint density $p(\boldsymbol{y}, \boldsymbol{f})$ with Pólya-Gamma variables and obtain

$$p(\boldsymbol{y}, \boldsymbol{f}, \boldsymbol{\omega}) \propto \exp\left(\frac{1}{2}\boldsymbol{y}^\top \boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top \Omega \boldsymbol{f}\right) p(\boldsymbol{f})p(\boldsymbol{\omega}), \tag{3.4}$$

where $\Omega = \mathrm{diag}(\boldsymbol{\omega})$ is the diagonal matrix of the Pólya-Gamma variables $\{\omega_i\}$.

Interestingly, employing a structured mean field variational inference approach (cf. Section 3.3) to the plain Pólya-Gamma augmented model (3.4) leads to the same bound for GP classification derived by Gibbs and MacKay (2000). This is an interesting new perspective on this bound since they do not employ a data augmentation approach. We provide a proof in Appendix A.3. Our approach goes beyond Gibbs and MacKay (2000) by providing a fully Bayesian perspective, including a sparse GP prior in the model and proposing a scalable inference algorithm based on natural gradients (Section 3.3).

3.3 Inference

To scale our model to big datasets, we approximate the latent GP $\boldsymbol{f}$ by a *sparse GP* building on *inducing points*. As outlined in Section 1.2, we introduce M inducing points $\boldsymbol{u}$ and connect the GP values with the inducing points via the joint prior distribution $p(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})$ given in Eq. 1.5.

We follow a structured mean field approach (Wainwright and Jordan, 2008) and assume independence between the inducing variables $\boldsymbol{u}$ and Pólya-Gamma variables $\boldsymbol{\omega}$, yielding a variational distribution of the form $q(\boldsymbol{u}, \boldsymbol{\omega}) = q(\boldsymbol{u})q(\boldsymbol{\omega})$. Setting the functional derivative of $\mathcal{L}$ w.r.t. $q(\boldsymbol{u})$ and $q(\boldsymbol{\omega})$ to zero, respectively, results in the following consistency condition for the maximum,

$$q(\boldsymbol{u}, \boldsymbol{\omega}) = q(\boldsymbol{u}) \prod_i q(\omega_i), \tag{3.5}$$

with $q(\omega_i) = \mathrm{PG}(\omega_i|1, c_i)$ and $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \Sigma)$. Remarkably, we do not have to use the full Pólya-Gamma class $\mathrm{PG}(\omega_i|b_i, c_i)$, but instead, consider the restricted class $b_i = 1$ since it already contains the optimal distribution. This can be easily seen by starting with a free-form mean field variational inference approach and deriving the optimal condition $b_i = 1$.

Using (3.5) as variational family, which is parameterized by the variational parameters $\{\boldsymbol{\mu}, \Sigma, \boldsymbol{c}\}$ leads to a closed-form expression of the variational bound

$$\begin{aligned}
&\mathcal{L}(\boldsymbol{c}, \boldsymbol{\mu}, \Sigma) \\
&= \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{f})] - \mathrm{KL}\left(q(\boldsymbol{u}, \boldsymbol{\omega})||p(\boldsymbol{u}, \boldsymbol{\omega})\right) \\
&\overset{\mathsf{c}}{=} \frac{1}{2}\Bigg(\log|\Sigma| - \log|K_{mm}|) - \mathrm{tr}(K_{mm}^{-1}\Sigma) - \boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} \\
&\quad + \sum_i \Big\{ y_i \boldsymbol{\kappa}_i \boldsymbol{\mu} - \theta_i \left(\widetilde{K}_{ii} - \boldsymbol{\kappa}_i \Sigma \boldsymbol{\kappa}_i^\top - \boldsymbol{\mu}^\top \boldsymbol{\kappa}_i^\top \boldsymbol{\kappa}_i \boldsymbol{\mu} \right) \\
&\quad + c_i^2 \theta_i - 2 \log\cosh \frac{c_i}{2} \Big\} \Bigg),
\end{aligned} \tag{3.6}$$

where $\theta_i = \frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)$ and $\boldsymbol{\kappa}_i = K_{im}K_{mm}^{-1}$. Remarkably, all intractable terms involving expectations of $\log \mathrm{PG}(\omega_i|1, 0)$ cancel out. Details are provided in Appendix A.1.

Stochastic variational inference. Our algorithm alternates between updates of the local variational parameters $\boldsymbol{c}$ and global parameters $\boldsymbol{\mu}$ and Σ. In each iteration, we update the parameters based on a mini-batch of the data $\mathcal{S} \subset \{1, \ldots, n\}$ of size $s = |\mathcal{S}|$.

We update the *local parameters* $\boldsymbol{c}_{\mathcal{S}}$ in the mini-batch $\mathcal{S}$ by employing coordinate ascent. To this end, we fix the global parameters and analytically compute the unique maximum of (3.6) w.r.t. the local parameters, leading to the updates

$$c_i = \sqrt{\widetilde{K}_{ii} + \boldsymbol{\kappa}_i \Sigma \boldsymbol{\kappa}_i^\top + \boldsymbol{\mu}^\top \boldsymbol{\kappa}_i^\top \boldsymbol{\kappa}_i \boldsymbol{\mu}} \tag{3.7}$$

for $i \in \mathcal{S}$.

We update the *global parameters* by employing stochastic optimization of the variational bound (3.6). The optimization is based on stochastic estimates of the natural gradients of the global parameters. We use the natural parameterization of the variational Gaussian distribution, i.e., the parameters $\boldsymbol{\eta}_1 := \Sigma^{-1}\boldsymbol{\mu}$ and $\eta_2 = -\frac{1}{2}\Sigma^{-1}$. Using the natural parameters results in simpler and more effective updates. The natural gradients based on the mini-batch $\mathcal{S}$ are given by

$$\begin{aligned} \widetilde{\nabla}_{\boldsymbol{\eta}_1} \mathcal{L}_{\mathcal{S}} &= \frac{n}{2s} \boldsymbol{\kappa}_{\mathcal{S}}^\top \boldsymbol{y}_{\mathcal{S}} - \boldsymbol{\eta}_1 \\ \widetilde{\nabla}_{\eta_2} \mathcal{L}_{\mathcal{S}} &= -\frac{1}{2}\left(K_{mm}^{-1} + \frac{n}{s} \boldsymbol{\kappa}_{\mathcal{S}}^\top \Theta_{\mathcal{S}} \boldsymbol{\kappa}_{\mathcal{S}} \right) - \eta_2, \end{aligned} \tag{3.8}$$

where $\Theta = \operatorname{diag}(\boldsymbol{\theta})$ and $\theta_i = \frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)$. The factor $\frac{n}{s}$ is due to the rescaling of the mini-batches. The global parameters are updated according to a stochastic natural gradient ascent scheme. We employ the adaptive learning rate method described by Ranganath et al. (2013).

The natural gradient updates always lead to a positive definite covariance matrix[2] and in contrast to Hensman and Matthews (2015) our implementation does not require any assurance for positive-definiteness of the variational covariance matrix Σ. Details for the derivation of the updates can be found in Appendix A.2. The complexity of each iteration in the inference scheme is $\mathcal{O}(m^3)$, due to the inversion of the matrix η_2.

Predictions. The approximate posterior of the GP values and inducing variables is given by $q(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})$, where $q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \Sigma)$ denotes the optimal variational distribution. To predict the latent function values f_* at a test point x_* we substitute our approximate posterior into the standard predictive distribution

$$\begin{aligned} p(f_*|\boldsymbol{y}) &= \int p(f_*|\boldsymbol{f}, \boldsymbol{u}) p(\boldsymbol{f}, \boldsymbol{u}|\boldsymbol{y}) \mathrm{d}\boldsymbol{f} \mathrm{d}\boldsymbol{u} \\ &\approx \int p(f_*|\boldsymbol{f}, \boldsymbol{u}) p(\boldsymbol{f}|\boldsymbol{u}) q(\boldsymbol{u}) \mathrm{d}\boldsymbol{f} \mathrm{d}\boldsymbol{u} \\ &= \int p(f_*|\boldsymbol{u}) q(\boldsymbol{u}) \mathrm{d}\boldsymbol{u} = \mathcal{N}\left(f^*|\mu_*,\ \sigma_*^2\right), \end{aligned} \tag{3.9}$$

[2]This follows directly since K_{mm} and Θ are positive definite.

where the prediction mean and the variance is

$$\mu_* = \boldsymbol{K}_{*m} K_{mm}^{-1} \boldsymbol{\mu}$$
$$\sigma_*^2 = K_{**} + \boldsymbol{K}_{*m} K_{mm}^{-1} (\Sigma K_{mm}^{-1} - I) \boldsymbol{K}_{m*}.$$

The matrix $\boldsymbol{K}_{*m}$ denotes the kernel matrix between the test point and the inducing points and K_{**} the kernel value of the test point. The distribution of the test labels is easily computed by applying the logistic likelihood to (3.9),

$$p(y_* = 1|\boldsymbol{y}) = \int \sigma(f_*) p(f_*|\boldsymbol{y}) \mathrm{d}f_*. \tag{3.10}$$

This integral is analytically intractable but can be computed numerically by quadrature methods. This is adequate and fast since the integral is only one-dimensional.

Computing the mean and the variance of the predictive distribution has complexity $\mathcal{O}(m)$ and $\mathcal{O}(m^2)$, respectively.

Optimization of the hyperparameters. We select the optimal kernel hyperparameters h by maximizing the marginal likelihood $p(\boldsymbol{y}|h)$ via a variational expectation maximization approach. The details are described in Section 1.2.

3.4 Experiments

We compare our proposed method, efficient Gaussian process classification (X-GPC), with the state-of-the-art methods SVGPC (Salimbeni, Eleftheriadis, and Hensman, 2018), provided in the package GPflow[3] (Matthews et al., 2017), which builds on TensorFlow and the EP approach EPGPC by Hernández-Lobato and Hernández-Lobato (2016), implemented in R. All methods are applied to real-world datasets containing up to 11 million data points.

In all experiments, a squared exponential covariance function with a common length scale parameter for each dimension, an amplitude parameter and an additive noise parameter is used. The kernel hyperparameters are initialized to the same values and optimized using Adam (Kingma and Ba, 2015), while inducing points location are initialized via k-means++ (Arthur and Vassilvitskii, 2007) and kept fixed during training. The SVI based methods, X-GPC and SVGPC, use an adaptive learning rate. All algorithms are run on a single CPU. We experiment on 12 datasets from the OpenML website and the UCI repository ranging from 768 to 11 million data points.

[3]We use GPflow version 1.2.0.

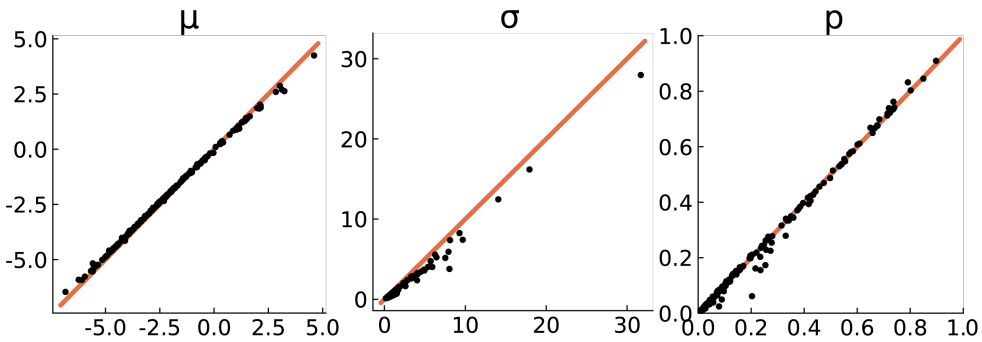

Figure 3.1: Posterior mean (μ), variance (σ) and predictive marginals (p) on the Diabetes dataset. Each plot shows the MCMC ground truth on the x-axis and the estimated value of our model on the y-axis. Our approximation is very close to the ground truth.

In the first experiment (Section 3.4.1), we examine the quality of the approximation provided by X-GPC. In the next experiment, we evaluate the prediction performance and run time of X-GPC, and SVGPC and EPGPC on several real-world datasets. Finally, in Section 3.4.3, we examine the sensitivity of all methods to the number of inducing points.

3.4.1 Quality of the approximation

We empirically examine the quality of the variational approximation provided by our method. In Fig. 3.1, we compare the approximations to the true posterior obtained by employing an asymptotically exact Gibbs sampler as described in Section 2.4 and also discussed by Polson and Scott, 2011; Linderman, Johnson, and Adams, 2015. We compare the posterior mean and variance as well as the prediction probabilities with the ground truth. Since the Gibbs sampler does not scale to large datasets we experiment on the small Diabetes dataset ($n = 768$). In Fig. 3.1 we plot the approximated values vs. the ground truth. We find that our approximation is very close to the true posterior.

3.4.2 Numerical comparison

We evaluate the prediction performance and run time of our method X-GPC and the competing methods SVGPC and EPGPC. We experiment on a variety of different datasets and report the resulting prediction error, negative test log-likelihood and run time for each method in Table 3.1.

The experiments are conducted as follows. For each dataset, we perform a 10-fold cross-validation and for datasets with more than one million points, we limit the

Dataset		**X-GPC**	SVGPC	EPGPC
aXa	Error	$\mathbf{0.17 \pm 0.07}$	$\mathbf{0.17 \pm 0.07}$	$\mathbf{0.17 \pm 0.07}$
$n = 36,974$	NLL	$\mathbf{0.29 \pm 0.13}$	0.36 ± 0.13	0.34 ± 0.13
$d = 123$	Time	$\mathbf{47 \pm 2.2}$	451 ± 7.8	214 ± 4.8
Bank Market.	Error	0.14 ± 0.12	$\mathbf{0.12 \pm 0.12}$	$\mathbf{0.12 \pm 0.13}$
$n = 45,211$	NLL	$\mathbf{0.27 \pm 0.22}$	0.31 ± 0.26	0.33 ± 0.20
$d = 43$	Time	$\mathbf{9 \pm 1.5}$	205 ± 6.6	46 ± 3.5
Click Pred.	Error	$\mathbf{0.17 \pm 0.00}$	$\mathbf{0.17 \pm 0.00}$	$\mathbf{0.17 \pm 0.01}$
$n = 399,482$	NLL	$\mathbf{0.39 \pm 0.07}$	0.46 ± 0.00	0.46 ± 0.01
$d = 12$	Time	$\mathbf{4.5 \pm 1.3}$	102 ± 3.0	8.1 ± 0.45
Cod RNA	Error	$\mathbf{0.04 \pm 0.00}$	$\mathbf{0.04 \pm 0.00}$	$\mathbf{0.04 \pm 0.00}$
$n = 343,564$	NLL	$\mathbf{0.11 \pm 0.03}$	0.13 ± 0.00	0.12 ± 0.00
$d = 8$	Time	$\mathbf{3.7 \pm 0.13}$	115 ± 4.3	869 ± 5.2
Diabetes	Error	$\mathbf{0.23 \pm 0.07}$	$\mathbf{0.23 \pm 0.06}$	0.24 ± 0.06
$n = 768$	NLL	$\mathbf{0.47 \pm 0.11}$	0.47 ± 0.10	0.48 ± 0.09
$d = 8$	Time	8.8 ± 0.12	150 ± 5.1	$\mathbf{8 \pm 0.45}$
Electricity	Error	$\mathbf{0.24 \pm 0.06}$	0.26 ± 0.06	0.26 ± 0.06
$n = 45,312$	NLL	$\mathbf{0.31 \pm 0.17}$	0.53 ± 0.08	0.53 ± 0.06
$d = 8$	Time	$\mathbf{8.2 \pm 0.48}$	356 ± 6.9	13.5 ± 1.50
German	Error	$\mathbf{0.25 \pm 0.12}$	$\mathbf{0.25 \pm 0.11}$	0.26 ± 0.13
$n = 1,000$	NLL	$\mathbf{0.44 \pm 0.17}$	0.51 ± 0.15	0.53 ± 0.11
$d = 20$	Time	17 ± 0.42	374 ± 7.3	$\mathbf{5.2 \pm 0.03}$
Higgs	Error	$\mathbf{0.33 \pm 0.01}$	0.45 ± 0.01	0.38 ± 0.01
$n = 11,000,000$	NLL	$\mathbf{0.55 \pm 0.13}$	0.69 ± 0.00	0.66 ± 0.00
$d = 28$	Time	$\mathbf{23 \pm 0.88}$	294 ± 54	8732 ± 867
IJCNN	Error	0.03 ± 0.01	0.06 ± 0.01	$\mathbf{0.02 \pm 0.01}$
$n = 141,691$	NLL	0.10 ± 0.03	0.15 ± 0.07	$\mathbf{0.09 \pm 0.04}$
$d = 22$	Time	$\mathbf{17 \pm 0.44}$	1033 ± 45	756 ± 8.6
Mnist	Error	0.14 ± 0.01	0.44 ± 0.13	$\mathbf{0.12 \pm 0.01}$
$n = 70,000$	NLL	$\mathbf{0.24 \pm 0.10}$	0.66 ± 0.11	0.27 ± 0.01
$d = 780$	Time	$\mathbf{200 \pm 5.5}$	991 ± 23	806 ± 5.2
Shuttle	Error	$\mathbf{0.01 \pm 0.01}$	$\mathbf{0.01 \pm 0.00}$	$\mathbf{0.01 \pm 0.01}$
$n = 58,000$	NLL	$\mathbf{0.07 \pm 0.01}$	$\mathbf{0.07 \pm 0.00}$	$\mathbf{0.07 \pm 0.01}$
$d = 9$	Time	$\mathbf{0.01 \pm 0.00}$	7.5 ± 0.7	100 ± 0.63
SUSY	Error	$\mathbf{0.21 \pm 0.00}$	0.22 ± 0.00	0.22 ± 0.00
$n = 5,000,000$	NLL	$\mathbf{0.31 \pm 0.10}$	0.49 ± 0.01	0.50 ± 0.00
$d = 18$	Time	$\mathbf{14 \pm 0.29}$	$10,000$	$10,000$
wXa	Error	$\mathbf{0.03 \pm 0.01}$	0.04 ± 0.01	$\mathbf{0.03 \pm 0.01}$
$n = 34,780$	NLL	0.27 ± 0.07	0.25 ± 0.07	$\mathbf{0.19 \pm 0.06}$
$d = 300$	Time	66 ± 16	612 ± 11	$\mathbf{1.4 \pm 0.10}$

Table 3.1: Average test prediction error, negative test log-likelihood (NLL) and time in seconds along with one standard deviation. Best values are highlighted.

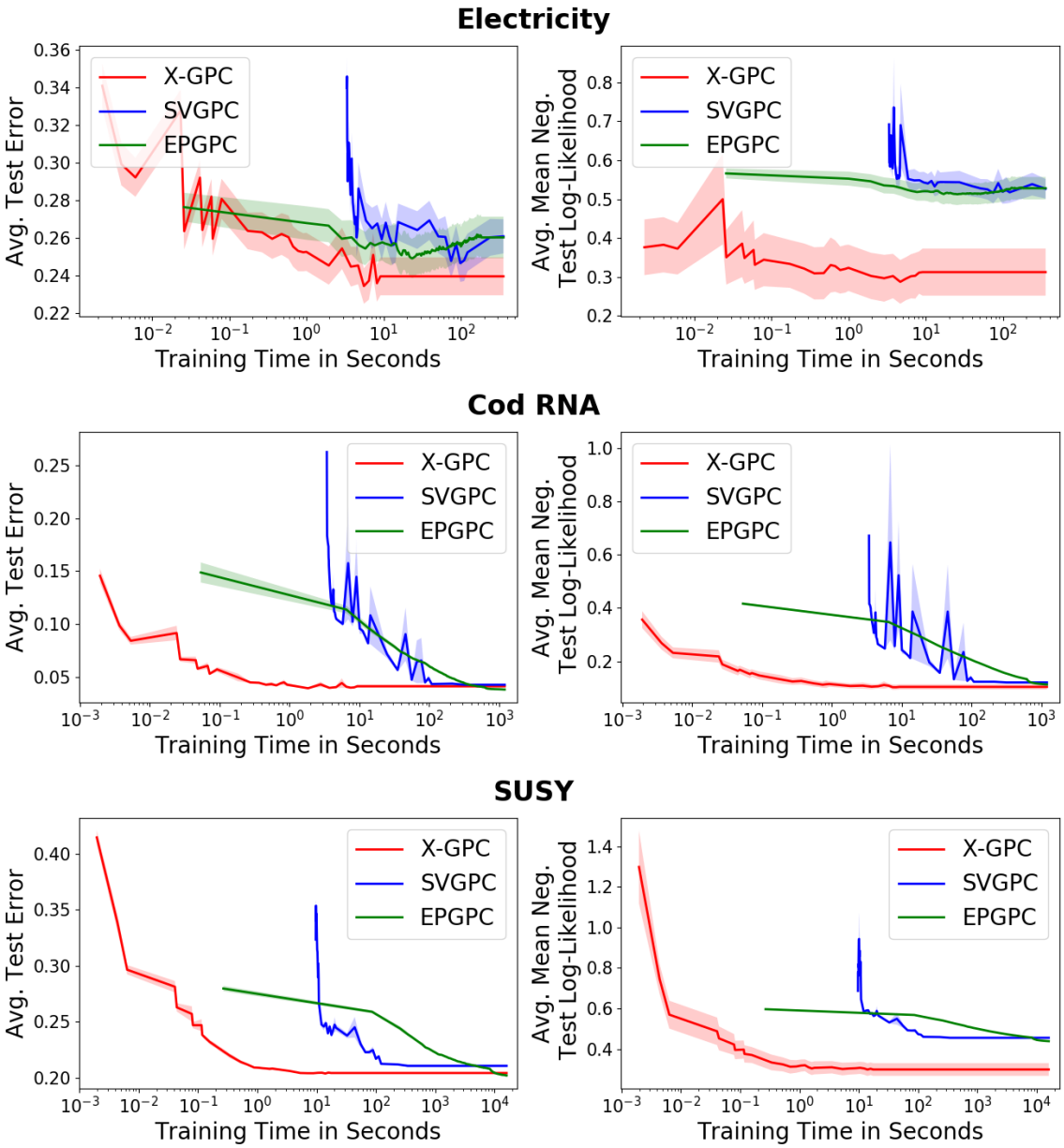

Figure 3.2: Average negative test log-likelihood and average test prediction error as a function of training time (seconds in a $\log_{10}$ scale) on the datasets Electricity (45,312 points), Cod RNA (343,564 points) and SUSY (5 million points). X-GPC (proposed) reaches values close to the optimum after only a few iterations, whereas SVGPC and EPGPC are one to two orders of magnitude slower.

test set to 100,000 points. We report the average prediction error, the negative test log-likelihood (3.10) and the run time along with one standard deviation. For all datasets, we use 100 inducing points and a mini-batch size of 100 points.

For X-GPC, we find that the following simple convergence criterion on the global parameters leads to good results: a sliding window average being smaller than a threshold of 10^{-4}. Unfortunately, the original implementations of SVGPC and EPGPC do not include a convergence criterion. We find that the trajectories of the global parameters of SVGPC tend to be noisy, and using a convergence criterion on the global parameters often leads to poor results. To have a fair comparison, we therefore monitor the convergence of the prediction performance on a hold-out set and use a sliding window average of size 5 and threshold 10^{-3} as a convergence criterion for all methods.

We observe that X-GPC is about one to two orders of magnitude faster than SVGPC and EPGPC on most datasets. Only on the dataset wXa, EPGPC is slightly faster than X-GPC. The prediction error is similar for all methods but X-GPC outperforms the competitors in terms of the test log-likelihood on most datasets (aXa, Bank Marketing, Click Prediction, Cod RNA, Diabetes, Electricity, German, Higgs, Mnist, SUSY). This means that the confidence levels in the predictions are better calibrated for X-GPC, i.e. when predicting a wrong label SVGPC and EPGPC tend to be more confident than X-GPC.

Performance as a function of time. Since all considered methods are based on optimization schemes, there is a trade-off between the run time of the algorithm and the prediction performance. We make this trade-off transparent by plotting the prediction performance as a function of time on each dataset. For each method, we employ a 10-fold cross-validation, and monitor the average negative test log-likelihood and prediction error on a hold-out test set as a function of time.

The results are displayed in Fig. 3.2 for three selected datasets, while the results for the remaining datasets are deferred to Appendix A.4. For all datasets we observe that after a few iterations X-GPC is already close to the optimum due to its efficient closed-form natural gradient updates. Both the prediction error and test log-likelihood converge around one to two orders of magnitude faster for X-GPC than for SVGPC and EPGPC. Moreover, the performance curves tend to be noisier for SVGPC than for X-GPC and EPGPC. For the datasets HIGGS and IJCNN, EPGPC lead to slightly better final prediction performance, but with the cost of a runtime being two to three orders of magnitude slower than X-GPC (EPGPC runs approx. 2.5 hours on the

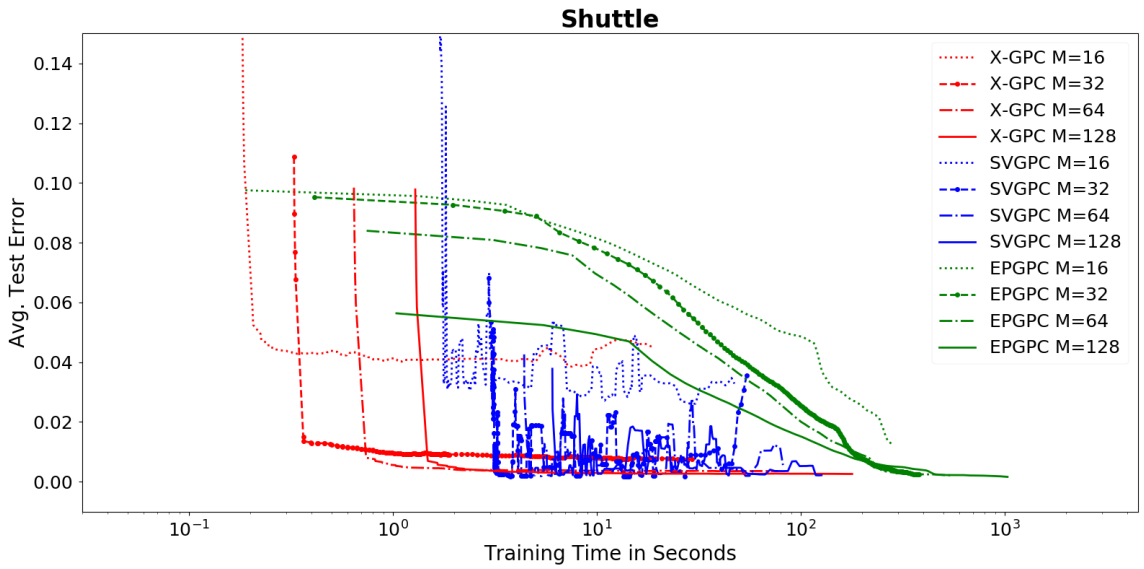

Figure 3.3: Prediction error as a function of training time (on a $\log_{10}$ scale) for the Shuttle dataset. Different numbers of inducing points are considered, $M = 16, 32, 64, 128$. X-GPC (proposed) converges the fastest in all settings of different numbers of inducing points. Using only 32 inducing points is enough for obtaining almost optimal prediction performance for all methods, but SVGPC becomes unstable in settings of less than 128 inducing points.

HIGGS dataset and X-GPC runs 23 seconds, and on the IJCNN dataset, EPGPC runs 756 seconds, whereas X-GPC runs 17 seconds).

All three methods are implemented in different programming frameworks: X-GPC in Julia, SVGPC in TensorFlow and EPGPC in R leading to different efficient implementations. However, we find that the main speed-up of our method is due to the efficient natural gradient updates and is only marginally related to the usage of a different programming language. To check this, we implemented EPGPC also in Julia and obtained similar runtimes. Since SVGPC is part of the already highly optimized GPflow package we only used the original implementation.

3.4.3 Inducing points

We examine the effect of different numbers of inducing points on the prediction performance and run time. For all methods, we compare different numbers of inducing points: $M = 16, 32, 64, 128$. For each setting, we perform a 10-fold cross-validation on the Shuttle dataset and plot the mean prediction error as a function of time. The results are displayed in Fig. 3.3. We observe that the higher the number of inducing

points, the better the prediction performance, but the longer the run time. Throughout all settings, our method is consistently faster of around one to two orders of magnitude than the competitors. On the Shuttle dataset using only $M = 32$ inducing points is enough and can only be marginally improved by using more inducing point for all methods. However, the performance curves of SVGPC are unstable when using less than 128 inducing points.

Chapter 4

Multi-class Gaussian Process Classification

In this chapter, we extend the ideas from the previous chapter, which concerned binary GP classification to the multi-class classification setting. We propose a novel modified softmax likelihood function, which has two benefits: it leads to well-calibrated uncertainty estimates and allows for an efficient latent variable augmentation. The augmented model has the advantage that it is conditionally conjugate leading to a fast variational inference method via block coordinate ascent updates. Previous approaches suffered from a trade-off between uncertainty calibration and speed. Our experiments show that our method leads to well-calibrated uncertainty confidences and competitive predictive performance while being up to two orders faster than the state of the art. The code is available via Github[1].

This chapter is based on:

> T. Galy-Fajou*, F. Wenzel*, C. Donner, M. Opper (2019). "Multi-Class Gaussian Process Classification Made Conjugate: Efficient Inference via Data Augmentation". In: *Conference on Uncertainty in Artificial Intelligence.*

In real-world decision making systems, it is important that classification methods do not only provide accurate predictions but also indicate when they are likely to be incorrect. Calibrated confidence estimates are important in many application domains such as self-driving cars (Bojarski et al., 2016), medical diagnosis (Caruana et al., 2015) and speech recognition (Xiong et al., 2016).

[1] `https://github.com/theogf/AugmentedGaussianProcesses.jl`
*Equal contributions.

In multi-class classification tasks, modern deep neural networks achieve state-of-the-art accuracies but often suffer from bad calibration (Guo et al., 2017). Gaussian process (GP) models provide an attractive alternative approach to multi-class classification problems.

Due to the Bayesian treatment of uncertainty, GPs have the advantage of leading to well-calibrated uncertainty estimates (Williams and Barber, 1998; Rasmussen and Williams, 2006). Furthermore, GP models become more expressive as the number of data points grows and allow for incorporating prior knowledge by using different kernel functions. However, inference in multi-class GP classification models is challenging.

In Chapter 3, we have shown that an efficient inference method can be developed for *binary* classification. The *multi-class* problem is more complicated because it involves not only one latent GP, but one GP for each class. In the common multi-class likelihoods, as e.g. the softmax function, the GPs are coupled. This leads to complicated multivariate integrals, which make a direct application of variational inference techniques intractable. Previous inference methods for the softmax model rely on approximations and do not scale (Williams and Barber, 1998; Chai, 2012).

To tackle this issue, Hernández-Lobato, Hernández-Lobato, and Dupont (2011) propose an alternative to the softmax, the *robust-max* likelihood. This likelihood simplifies the problem by focusing mainly on the maximal latent GP and discarding information of the other less likely classes. The model is robust against outliers and often yields good classification accuracy. However, it sacrifices the gradual response of the traditional softmax for an all-or-nothing criterion leading to bad uncertainty quantification.

In problems with well separated classes and a few outliers, the robust-max likelihood is an excellent choice, while in problems with overlapping classes a gradual classification criterion is more desirable (Xiong, Wu, and Liu, 2010). In this work, we introduce a novel likelihood, the *logistic-softmax* likelihood, which combines the best of both worlds. It has a gradual classification criterion similar to the traditional softmax, but on the other hand, also enables fast inference.

We propose an augmentation approach that renders the model conditionally conjugate. Inference in the augmented model is much easier. We derive a fast *variational inference* algorithm based on closed-form updates. Our inference approach is faster and more stable than the state of the art since it uses efficient block coordinate ascent updates and does not rely on sampling.

4.1 Background and related work

We begin our review by introducing the multi-class GP classification model. Related work can be grouped into approaches that consider alternative likelihood functions or apply data augmentation strategies.

Multi-class GP classification. We consider a dataset of N data points $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)^T$ with labels $\boldsymbol{y} = (y_1, \ldots, y_N)$, where $y_i \in \{1, \ldots, C\}$ and C is the total number of classes. The multi-class GP classification model consists of a latent GP prior for each class $\boldsymbol{f} = (f^1, \ldots, f^C)$, where $f^c \sim \mathrm{GP}(0, k^c)$ and k^c is the corresponding kernel function. The labels are modeled by a categorical likelihood

$$p(y_i = k|\boldsymbol{x}_i, \boldsymbol{f}_i) = g^k(\boldsymbol{f}(\boldsymbol{x}_i)), \tag{4.1}$$

where $g(.)$ is a function that maps the vector $\boldsymbol{f}_i$, containing the GP values for each class for the data point $\boldsymbol{x}_i$, to a probability vector.

The most common way to form a categorical likelihood is through the softmax transformation

$$p(y_i = k|\boldsymbol{f}_i) = \frac{\exp\left(f_i^k\right)}{\sum_{c=1}^{C} \exp\left(f_i^c\right)}, \tag{4.2}$$

where we use the shorthand $f_i^c = f^c(x_i)$ and for the sake of clarity we omit the conditioning on x_i.

There have been several early works addressing multi-class GP classification with a softmax likelihood (Williams and Barber, 1998; Kim and Ghahramani, 2006; Chai, 2012; Riihimäki, Jylänki, and Vehtari, 2013). Nevertheless, these methods do not scale well with the number of data points. Izmailov, Novikov, and Kropotov (2018) employ tensor train decomposition to use high numbers of inducing points but do not provide efficient closed-form updates.

The robust-max likelihood. Recently, there have been advances to scale multi-class GP classification to big datasets by changing the likelihood. Hernández-Lobato, Hernández-Lobato, and Dupont (2011) propose the *robust-max* likelihood

$$p(y = k|\boldsymbol{f}) = (1 - \epsilon) \prod_{c \neq y}^{C} \Theta\left(f^k - f^c\right) + \frac{\epsilon}{C}, \tag{4.3}$$

where ϵ is the probability of a labeling error and Θ is the Heaviside function. This likelihood simplifies the problem as it leads to a decoupling of the latent GPs.

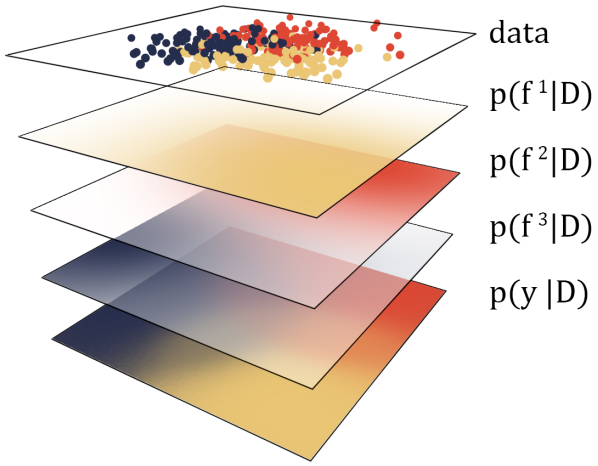

Figure 4.1: In a GP multi-class classification model, each class density is modeled by an individual GP $p(f^c|D)$. For predictions $p(y|D)$, the latent GPs are marginalized out.

Originally, the authors propose an expectation propagation (EP) based approach, which only scales to small datasets. Hensman et al. (2015) and Salimbeni, Eleftheriadis, and Hensman (2018) scale this model to big datasets employing a variational inference approach but rely on numerical quadrature. As we show later, this likelihood has the big disadvantage of leading to poor confidence calibration.

The Heaviside likelihood. Villacampa-Calvo and Hernández-Lobato (2017) build on the Heaviside likelihood

$$p(y = k|\boldsymbol{f}) = \prod_{c \neq y}^{C} \Theta\left(f^k - f^c\right), \tag{4.4}$$

where Θ is again the Heaviside function. The authors propose a scalable expectation propagation approach but have to make approximations on the likelihood. The inference is still slow and the applicability to big datasets is limited.

Data augmentation. Linderman, Johnson, and Adams (2015) consider data augmentation for multinomial likelihoods but focus on sampling. The approach has the disadvantage of breaking the symmetry between the classes and is limited to small datasets. Polson, Scott, and Windle (2013) propose conditionally conjugate Pólya-Gamma augmentation for the softmax likelihood (extended by Cesnovar and Strumbelj (2017) to GPU support), which is suitable for sampling but cannot be used for obtaining an efficient variational inference algorithm since the ELBO is intractable.

Girolami and Rogers (2006) propose an augmentation strategy to multinomial probit regression but do not scale. Ruiz et al. (2018) propose an augmentation approach for enabling subsampling of classes for parametric models with categorical likelihoods. The approach is limited to parametric models and cannot be applied to GP models.

4.2 Conjugate multi-class Gaussian process classification

We formulate a multi-class GP classification model, which leads to well calibrated confidences and is amenable to fast inference. We define a new likelihood function, termed the *logistic-softmax* , which shares the good prediction properties of the softmax. But in addition, it has the advantage that it allows for a data augmentation approach that renders the model conditionally conjugate. The augmented posterior can then be efficiently approximated by a structured mean field variational inference method resulting in a fast algorithm with closed-form updates.

4.2.1 The logistic-softmax GP model

We consider the multi-class GP classification model as described in Eq. 4.1. Different functions g for mapping real vectors to probability vectors that have been considered in literature include the softmax (Eq. 4.2), the multinomial probit (Albert and Chib, 1993), the robust-max likelihood (Eq. 4.3) and the Heaviside likelihood (Eq. 4.4).

In this work, we propose the *logistic-softmax*:

$$p(y_i = k|\boldsymbol{f}_i) = \frac{\sigma\left(f_i^k\right)}{\sum_{c=1}^{C} \sigma\left(f_i^c\right)}, \tag{4.5}$$

where $\sigma(z) = (1 + \exp(-z))^{-1}$ is the logistic function. Our likelihood is a modified version of the softmax likelihood that replaces the inner exponential functions by logistic functions. Alternatively, it can be interpreted as the standard softmax applied to a non-linearly transformed GP, i.e. $p(y_i|\boldsymbol{f}_i) = \text{softmax}(\log \sigma(\boldsymbol{f}_i))$. The likelihood reduces to the binary logistic likelihood for $C = 2$.

In the following section, we derive a three steps augmentation scheme, where we (i) decouple the GP latent variables f_i^k in the denominator by the introduction of a set of auxiliary λ-variables, (ii) further simplify the model likelihood by introducing Poisson random variables, and finally (iii) use a Pólya–Gamma representation of the sigmoid function (Polson, Scott, and Windle, 2013) to achieve the desired conjugate representation of the model.

4.2.2 Towards a conjugate augmentation

We expand the logistic-softmax likelihood (4.5) by three data augmentation steps leading to a conditionally conjugate model. The final model is displayed in Fig. 4.2. In the following, we present the augmentations.

Augmentation 1: Gamma augmentation. To remedy the intractable normalizer term we make use of the integral identity $\frac{1}{z} = \int_0^\infty \exp(-\lambda z)\mathrm{d}\lambda$ and express the likelihood (4.5) as

$$\begin{aligned} p(y_i = k|\boldsymbol{f}_i) &= \frac{\sigma\left(f_i^k\right)}{\sum_{c=1}^C \sigma\left(f_i^c\right))} \\ &= \sigma\left(f_i^k\right) \int_0^\infty \exp\left(-\lambda_i \sum_{c=1}^C \sigma\left(f_i^c\right)\right) \mathrm{d}\lambda_i. \end{aligned}$$

This augmentation is well known in the Gibbs sampling community to deal with intractable normalization constants (see e.g. Walker (2011)) but is not often used in the setting of variational inference. By interpreting λ_i as an additional latent variable we obtain the augmented likelihood

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i) = \sigma(f_i^k) \prod_{c=1}^C \exp\left(-\lambda_i \sigma(f_i^c)\right), \tag{4.6}$$

and we impose the improper prior $p(\lambda_i) \propto \mathbb{1}_{[0,\infty)}(\lambda_i)$. The improper prior is not problematic since it leads to a proper complete conditional distribution as we will see at the end of the section.

Augmentation 2: Poisson augmentation. We rewrite the exponential factors in (4.6) based on the moment generation function of the Poisson distribution $\mathrm{Po}(\cdot|\lambda)$, which is

$$\exp(\lambda(z-1)) = \sum_{n=0}^\infty z^n \mathrm{Po}(n|\lambda).$$

Using $z = \sigma(-f)$ and the fact that $\sigma(f) = 1 - \sigma(-f)$ we rewrite the exponential factors as

$$\begin{aligned} \exp\left(-\lambda_i \sigma(f_i^c)\right) &= \exp\left(\lambda_i(\sigma(-f_i^c) - 1)\right) \\ &= \sum_{n_i^c=0}^\infty (\sigma(-f_i^c))^{n_i^c} \mathrm{Po}(n_i^c|\lambda_i), \end{aligned}$$

which leads to the augmented likelihood

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i, \boldsymbol{n}_i) = \sigma(f_i^k) \prod_{c=1}^{C} (\sigma(-f_i^c))^{n_i^c}, \tag{4.7}$$

where $\boldsymbol{n}_i = (n_i^1, \ldots, n_i^C)$ and the augmented Poisson variables are distributed as $p(n_i^c|\lambda_i) = \text{Po}(n_i^c|\lambda_i)$, see e.g. Donner and Opper, 2017; Donner and Opper, 2018. Note that this augmentation is only possible since the transformation on f_i^c is bounded, hence the need for a modified likelihood.

Augmentation 3: Pólya-Gamma augmentation. In the last augmentation step, we aim for a Gaussian representation of the sigmoid function. The Pólya-Gamma representation (Polson, Scott, and Windle, 2013) allows for rewriting the sigmoid function as a scale mixture of Gaussians

$$\sigma(z)^n = \int_0^\infty 2^{-n} \exp\left(\frac{nz}{2} - \frac{z^2}{2}\omega\right) \text{PG}(\omega|n, 0)\text{d}\omega, \tag{4.8}$$

where $\text{PG}(\omega|n, b)$ is a Pólya-Gamma distribution. Pólya-Gamma variables are well suited for augmentations since the moments are known analytically and an efficient sampler exists (Polson, Scott, and Windle, 2013). By applying this augmentation to (4.7) we obtain

$$\begin{aligned} &p(y_i = k|\boldsymbol{f}_i, \lambda_i, \boldsymbol{n}_i, \boldsymbol{\omega}_i) = \\ &\prod_{c=1}^{C} 2^{-(y_i'^c + n_i^c)} \exp\left(\frac{(y_i'^c - n_i^c) f_i^c}{2} - \frac{(f_i^c)^2}{2}\omega_i^c\right), \end{aligned} \tag{4.9}$$

where $\boldsymbol{\omega}_i = (\omega_i^1, \ldots, \omega_i^C)$ are Pólya-Gamma variables with distributions

$$p(\boldsymbol{\omega}_i|\boldsymbol{n}_i, y_i) = \prod_{c=1}^{C} \text{PG}(\omega_i^c|y_i'^c + n_i^c, 0),$$

where $\boldsymbol{y}'$ is an $N \times C$-dimensional one-hot encoding of the labels , i.e. $y_i'^c$ is 1 if $y_i = c$, and 0 otherwise. Details are deferred to Appendix B.1.

Realizing that (4.9) has a Gaussian form with respect to $\boldsymbol{f}_i$, we achieved our goal of a conjugate representation of the latent GPs. As we will show in the next paragraph the model is also conditionally conjugate for the augmented variables.

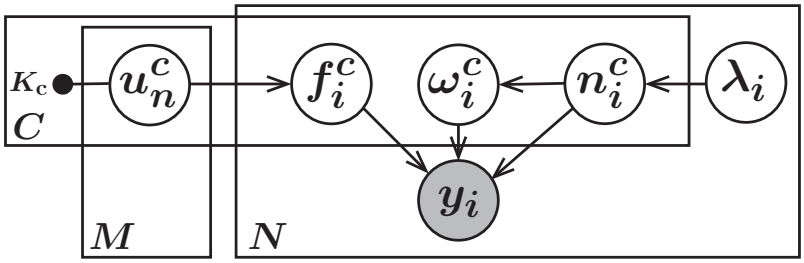

Figure 4.2: The final augmented model as presented in Section 4.2.2. Shaded circles represent observable variables, empty circles latent variables and dots hyperparameters.

The final model. The effort of the augmentations finally pays off as the final augmented model is now tractable and the complete conditional distributions are given in closed form.

The complete conditionals of the GPs $\boldsymbol{f}^c$ are

$$p(\boldsymbol{f}^c \mid \boldsymbol{y}, \boldsymbol{\omega}^c, \boldsymbol{n}^c) = \mathcal{N}\left(\boldsymbol{f}^c \mid \frac{1}{2}A^c(\boldsymbol{y}'^c - \boldsymbol{n}^c), A^c\right),$$

where the conditional covariance matrix is given by $A^c = (\mathrm{diag}(\boldsymbol{\omega}^c) + K_c^{-1})^{-1}$ and K_c is the kernel matrix of the GP $\boldsymbol{f}^c$. For the conditional distribution of $\boldsymbol{\lambda}$ we get

$$p(\lambda_i \mid \boldsymbol{n}_i) = \mathrm{Ga}\left(\lambda_i \mid 1 + \sum_{c=1}^{C} n_i^c, C\right),$$

where $\mathrm{Ga}(\cdot|a, b)$ denotes a gamma distribution with shape parameter a and rate parameter b. The improper prior on λ_i does not impose an issue since the complete conditional distribution is proper.

For the Poisson variables $\boldsymbol{n}$, we get

$$p(n_i^c \mid f_i^c, \lambda_i) = \mathrm{Po}\left(n_i^c \mid \lambda_i \sigma(f_i^c)\right),$$

Finally, for the Pólya-Gamma variables $\boldsymbol{\omega}$ the complete conditional distributions are

$$p(\omega_i^c \mid n_i^c, f_i^c, y_i) = \mathrm{PG}\left(\omega_i^c \mid y'^c_i + n_i^c, |f_i^c|\right).$$

4.3 Inference

We derive a variational approximation of the posterior of the augmented model (4.9). In the following, we develop an efficient stochastic variational inference (SVI) algorithm that is based on closed-form block coordinate ascent updates. Our method allows both for subsampling of data points and outcomes (classes) scaling to datasets with a large number of data points and a large number of classes.

4.3.1 Variational approximation

To scale our model to big datasets, we approximate the latent GPs $\boldsymbol{f}^c$ by *sparse GPs* building on *inducing points*. For each GP $\boldsymbol{f}^c$, we introduce M inducing points $\boldsymbol{u}^c$ and connect the GP values with the inducing points via the joint prior distribution $p(\boldsymbol{f}^c, \boldsymbol{u}^c)$ given in Section 1.2.

We approximate the posterior distribution of the latent sparse GPs $\boldsymbol{u}$ and the augmented variables $\boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}$ by assuming the following structure of the variational distribution $q(\boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}) = q(\boldsymbol{u}, \boldsymbol{\lambda})q(\boldsymbol{n}, \boldsymbol{\omega})$. Note that the only assumption on the variational posterior is the decoupling of two groups of variables. Since our model is conditionally conjugate, the family of the optimal variational distribution can be easily determined by averaging the complete conditionals in log-space (Blei, Kucukelbir, and McAuliffe, 2017). From the above decoupling assumption, it follows that the optimal variational posterior has a factorizing form $q(\boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{n}, \boldsymbol{\omega}) = q(\boldsymbol{u})q(\boldsymbol{\lambda})q(\boldsymbol{\omega}, \boldsymbol{n})$ and the factors are

$$
\begin{aligned}
q(\boldsymbol{u}) &= \prod_c \mathcal{N}(\boldsymbol{u}^c | \boldsymbol{\mu}^c, \Sigma^c), \; q(\boldsymbol{\lambda}) = \prod_i \mathrm{Ga}(\lambda_i | \alpha_i, \beta_i), \\
q(\boldsymbol{\omega}, \boldsymbol{n}) &= \prod_{i,c} \mathrm{PG}(\omega_i^c | {y'}_i^c + n_i^c, b_i^c) \mathrm{Po}(n_i^c | \gamma_i^c),
\end{aligned}
$$

where $\boldsymbol{\mu}^c$, Σ^c, α_i, β_i, b_i^c, γ_i^c, for all $i \in \{1, \ldots, N\}$ and $c \in \{1, \ldots, C\}$ are the *variational parameters*.

4.3.2 Inference method

Building on the conditionally conjugate representation of our model deriving efficient variational parameter updates is straightforward. We implement the SVI algorithm described by Hoffman et al. (2013), which builds on block coordinate ascent updates and is outlined in Section 2.2.

Applying the coordinate ascent update equation of standard stochastic variational inference scheme (see e.g. Hoffman et al., 2013 and Section 2.2) to each variational

distribution, we obtain the closed-form update for each variational parameter,

$$\begin{aligned}\overline{f_i^c} =& \sqrt{\mathbb{E}_{q(f^c)}\left[(f_i^c)^2\right]} \\ =& \sqrt{\widetilde{K}_{ii}^c + \kappa_i^c \Sigma^c \kappa_i^{c\top} + (\kappa_i^c \boldsymbol{\mu}^c)^\top \kappa_i^c \boldsymbol{\mu}^c}\end{aligned} \tag{4.10}$$

$$\gamma_i^c = \frac{\exp\left(\psi(\alpha_i)\right) \exp\left(-\frac{\kappa_i^c \boldsymbol{\mu}^c}{2}\right)}{\beta_i \cosh\left(\frac{\overline{f_i^c}}{2}\right)} \tag{4.11}$$

$$\alpha_i = 1 + \sum_{c=1}^{C} \gamma_i^c, \quad \beta_i = C \tag{4.12}$$

$$b_i^c = \overline{f_i^c}, \tag{4.13}$$

$$\theta_i^c = \mathbb{E}_{q(\omega_i^c, n_i^c)}\left[\omega_i^c\right] = \frac{y_i'^c + \gamma_i^c}{2b_i^c} \tanh \frac{b_i^c}{2} \tag{4.14}$$

$$\boldsymbol{\mu}^c = \frac{1}{2}(\Sigma^c)^{-1} \kappa^{c\top} \left(\boldsymbol{y}'^c - \boldsymbol{\gamma}^c\right) \tag{4.15}$$

$$\Sigma^c = \left(\kappa^{c\top} \mathrm{diag}\left(\boldsymbol{\theta}^c\right) \kappa^c + (K_{mm}^c)^{-1}\right)^{-1}, \tag{4.16}$$

where $\psi(.)$ is the digamma function. When $\kappa\mu \ll 0$, the update (4.11) easily overflows. The problem can be solved by approximating $\exp(-0.5\kappa\mu)/\cosh(0.5\bar{f}) \approx \sigma(\kappa\mu)$ by neglecting the variance terms $\widetilde{K} + \kappa\Sigma\kappa^\top$ for $\bar{f}$ (4.10).

Eq. 4.11 and Eq. 4.12 show a direct interdependence between α_i and γ_i^c. We apply an alternating inner loop optimization of both variables until convergence to solve the problem. We find that five iterations in the inner loop are sufficient.

When using mini-batches of the data, each global variational parameter (i.e. $\boldsymbol{\mu}^c$ and Σ^c) is updated using a convex combination of the old parameter and the CAVI update, which corresponds to a natural gradient ascent scheme (cf. Section 2.2).

The inference algorithm is summarized in Alg. 1 and its complexity is $\mathcal{O}(CM^3)$.

Extreme classification. When the number of possible outcomes (classes) C is very large, using probabilistic multi-class models becomes generally computationally expensive as the likelihood (categorical distribution) scales linearly with the number of classes. Using large categorical distributions is a challenging problem (Ruiz et al., 2018; Titsias, 2016).

Our method can deal with an extreme classification setting (large number of classes). In our augmentation, the GPs in the normalizer term are decoupled and allow for subsampling of the classes. This reduces the complexity to $\mathcal{O}(M^3)$, i.e. being independent of the number of classes.

Algorithm 1 Conjugate multi-class Gaussian process classification

Input: data $\boldsymbol{X}$,$\boldsymbol{y}$, mini-batch size $|\mathcal{S}|$
Output: variational posterior GPs $p(u^c|\mu^c, \Sigma^c)$
Set the learning rate schedules ρ_t, ρ_t^h appropriately
Initialize all variational parameters and hyperparameters
Select M inducing points locations (e.g. kMeans)
for iteration $t = 1, 2, \ldots$ **do**
    `# Sample mini-batch:`
    Sample a mini-batch of the data $\mathcal{S} \subset \{1, \ldots, N\}$
    `# Local variational updates`
    for $i \in \mathcal{S}$ **do**
        Update $(\alpha_i, \boldsymbol{\gamma}_i)$ (Eq. 4.11,4.12)
        for each class c **do**
            Update b_i^c (Eq. 4.13)
    `# Global variational GP updates`
    for each class c **do**
        $\mu^c \leftarrow (1 - \rho_t)\mu^c + \rho_t \hat{\mu}^c$ (Eq. 4.15)
        $\Sigma^c \leftarrow (1 - \rho_t)\Sigma^c + \rho_t \hat{\Sigma}^c$ (Eq. 4.16)
    `# Hyperparameter updates`
    Gradient step $h \leftarrow h + \rho_t^h \nabla_h \mathcal{L}$

We obtain the extreme classification version of our algorithm by a slight change to Alg. 1. The updates α_i are replaced by

$$\alpha_i = 1 + \frac{C}{|\mathcal{K}|} \sum_{c \in \mathcal{K}} \gamma_i^c, \tag{4.17}$$

where C is the number of classes and $|\mathcal{K}|$ is the number of sub-sampled classes. We display the extreme classification version of our algorithm in Appendix B.2. This approach is especially useful when using shared hyperparameters among the class-specific latent GPs.

Predictions. The posterior distribution of the latent function $p(f_*^c|\boldsymbol{x}_*, \boldsymbol{y})$ at a new test point $\boldsymbol{x}_*$ is approximated by

$$q(f_*^c|\boldsymbol{x}_*, \boldsymbol{y}) = \int p(f_*^c|\boldsymbol{u}^c) q(\boldsymbol{u}^c) \mathrm{d}\boldsymbol{u} = \mathcal{N}\left(f_*^c|{\mu_*}^c, {\sigma_*}^{2c}\right),$$

where the mean and the variance is

$$\begin{aligned} \mu_*^c &= \boldsymbol{K}_{*m}{}^c K_{mm}^{-1\ c} \boldsymbol{\mu}^c \\ \sigma_*^{2c} &= K_{**}{}^c + \boldsymbol{K}_{*m}{}^c K_{mm}^{-1\ c} (\Sigma^c K_{mm}^{-1\ c} - I) \boldsymbol{K}_{m*}{}^c. \end{aligned}$$

The matrix $\boldsymbol{K}_{*m}$ denotes the kernel matrix between the test point and the inducing points and K_{**} the kernel value of the test point. The final approximate predictive distribution of a test label is

$$p(y_* = k|\boldsymbol{x}_*, \boldsymbol{y}) \approx \int p(y_* = k|\boldsymbol{f}_*) \prod_{c=1}^{C} q(f_*^c|\boldsymbol{x}_*, \boldsymbol{y}) \mathrm{d}\boldsymbol{f}^*,$$

where $p(y_* = k|\boldsymbol{f}_*)$ is the logistic-softmax likelihood. This is a C-dimensional analytically intractable integral. We approximate it by Monte Carlo integration. For faster convergence, the random samples can be replaced by Quasi-Monte Carlo sequences (Owen, 1998; Buchholz, Wenzel, and Mandt, 2018). Finally, a point is classified by the highest predictive likelihood, $y_* = \arg\max_{c \in C} p\left(y_* = c \mid \boldsymbol{f}_*\right)$.

Optimization of the hyperparameters. We select the optimal kernel hyperparameters by maximizing the marginal likelihood $p(\boldsymbol{y}|h)$, where h denotes the set of hyperparameters. We apply the approximate expectation maximization approach explained in Section 2.2.

4.4 Experiments

In this section we empirically answer the following questions:

- What is the effect of using the softmax, *logistic-softmax*, robust-max and Heaviside likelihood on predictive performance and calibration quality? (Section 4.4.1)
- How does the augmentation affect the predictive performance? (Section 4.4.2)
- How does our method perform compared to other state-of-the-art GP based multi-class classification methods? (Section 4.4.4)

In all experiments, we use a squared exponential covariance function with automatic relevance determination (ARD): $k(\boldsymbol{x}, \boldsymbol{x'}) = \eta \exp\left(-\sum_{d=1}^{D} \frac{\left(x_d - x'_d\right)^2}{2l_d^2}\right)$, where we set the initial variance η to 1 and the length scales $\boldsymbol{l}$ are initialized to the median of the pairwise distance matrix of the data. The hyperparameters are optimized using Adam (Kingma and Ba, 2015). We use a collection of datasets from the LIBSVM repository[2]. Every dataset has been normalized to mean 0 and variance 1. For each method, we use 200 inducing points, unless stated otherwise. The initial inducing points locations are determined by the kmeans++ algorithm (Arthur and Vassilvitskii, 2007). We find

[2] `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/multiclass.html`

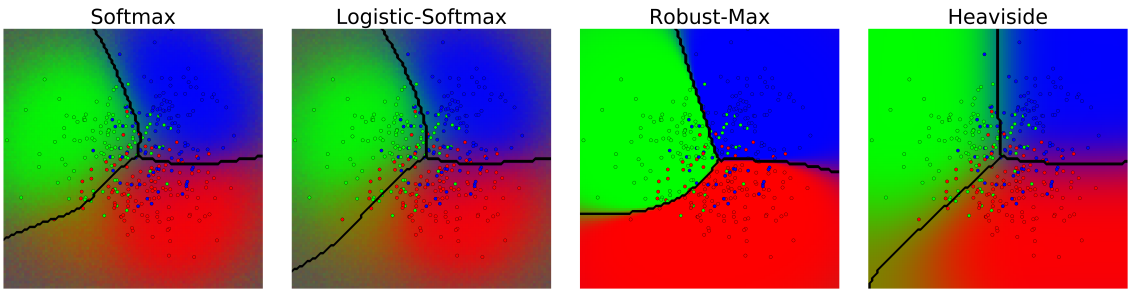

Figure 4.3: *Likelihood comparison:* Decision boundaries for each method on a toy dataset as described in Section 4.4.1 ($\sigma^2 = 0.5$). Each class is attributed to a color channel (red, green, blue) and predictive likelihoods are mapped into RGB values.

that fixing the locations while training gives good results. We use a mini-batch size of 200 and all experiments are performed on a single CPU.

4.4.1 Comparison of the different likelihoods

We begin the experiments by investigating the effect of using different likelihood functions. We compare our novel *logistic-softmax* (Eq. 4.5), the softmax (Eq. 4.2), the robust-max (Eq. 4.3) and the Heaviside likelihood (Eq. 4.4). For each model, we employ variational inference and obtain an approximate posterior. In this experiment, we apply variational inference directly to the non-augmented version of our *logistic-softmax* model $p(y, f)$. We focus on the non-augmented model to get a direct comparison of the effect of the different likelihoods and disable the additional influence of the augmentation. In this experiment, the gradients of the ELBO are estimated by sampling for all models.

To investigate uncertainty calibration, we create seven different toy datasets of 500 points with three classes. The data is generated from a mixture of Gaussians model with different variances σ^2. For $\sigma^2 = 0$, the classes are sharply separated and for $\sigma^2 = 1$, the classes highly overlap and are almost indistinguishable.

Visualization of the decision boundaries. For a better intuition of the behavior of each likelihood, we visualize the decision boundaries of the different methods. For this experiment, we use a toy dataset with moderate class overlap ($\sigma^2 = 0.5$). For each model, we map the predictive values of each class f^c to a RGB color channel (where each class corresponds to one color and mixing of colors indicates a contribution of multiple classes). A highly saturated color corresponds to high confidence in the class prediction, while mixed colors indicate zones of transition between classes and

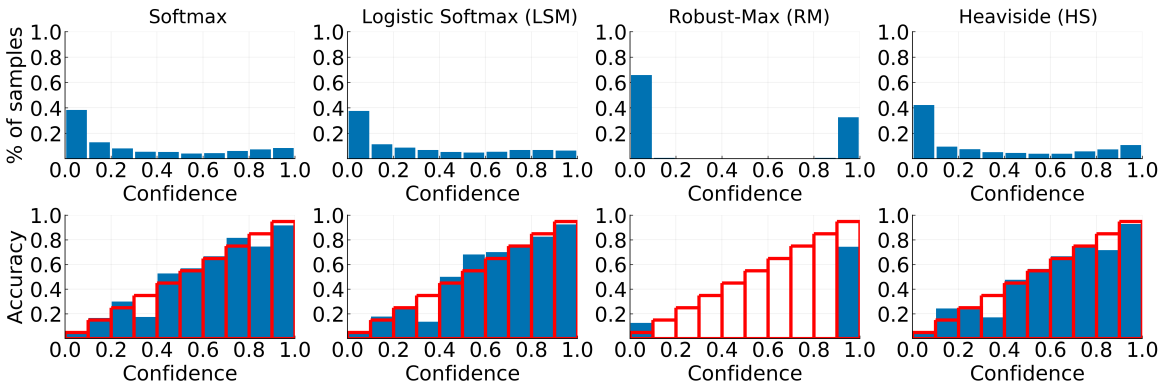

Figure 4.4: *Likelihood comparison:* Confidence histograms (top) and reliability diagrams (bottom) for four different likelihood models. The robust-max model always predicts with probability either close to one or close to zero leading to a poor confidence calibration.

lower confidence. The results are shown in Fig. 4.3. As expected, the robust-max likelihood leads to extremely sharp decision boundaries and high confidences for all regions (even for the regions where the data points overlap). The other likelihoods lead to better calibration resulting in soft boundaries and less confident predictions in the overlapping regions.

Uncertainty calibration. In Fig. 4.5, we plot test error, negative log-likelihood and calibration error as a function of the noise in the data. The (expected) calibration error is a summary statistic of calibration and is computed by the expectation between confidence and accuracy in the reliability diagram (cf. Guo et al., 2017).

For datasets where the classes are sharply separated (small σ^2), all models perform similarly. But for datasets where classes overlap (high σ^2), the robust-max performs poorly due to bad uncertainty calibration.

In Fig. 4.4, we show the confidence histograms and reliability diagrams for one dataset ($\sigma^2 = 0.5$). The diagrams are generated according to Naeini, Cooper, and Hauskrecht (2015) and Guo et al. (2017)—the reliability diagram displays the accuracy as a function of confidence (a perfectly calibrated model would produce the identity function) and the confidence histogram shows the empirical distribution of the prediction confidence.

The robust-max model fails to provide sensitive uncertainty estimates and only predicts with either probability close to zero or close to one. The softmax, *logistic-softmax* and Heaviside likelihood yield similar predictive performance and confidence calibration. However, as the following experiments show, our approach is much faster

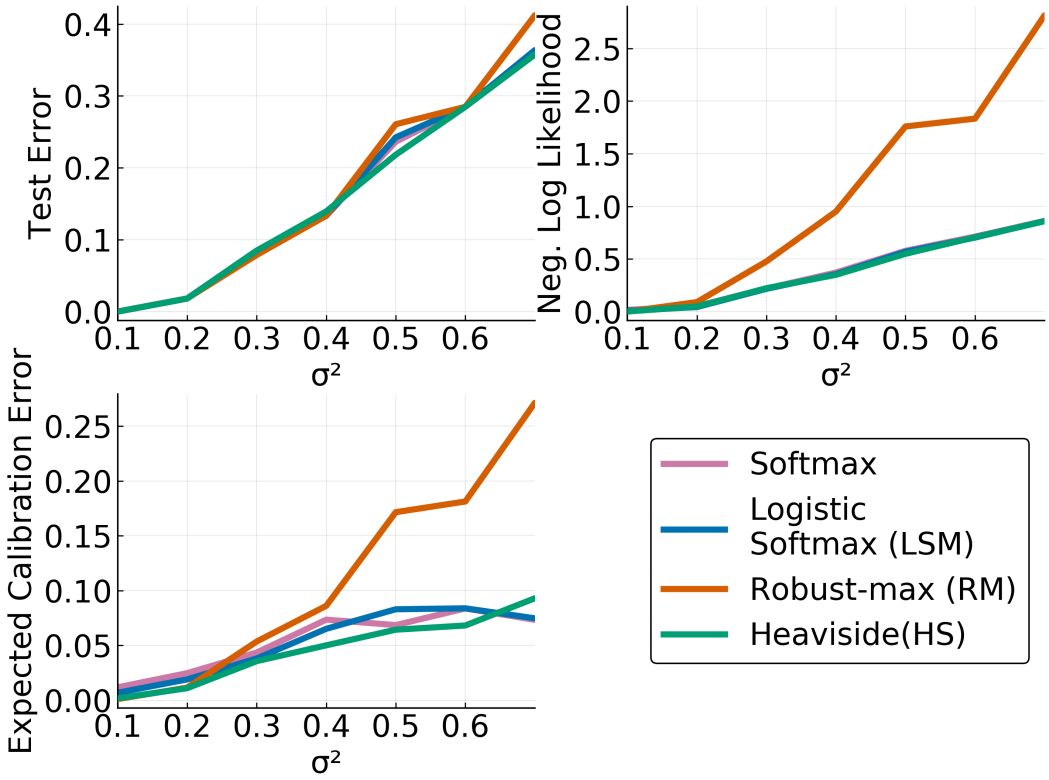

Figure 4.5: *Likelihood comparison:* The test error, negative log-likelihood and calibration error are plotted as a function of the noise (σ^2) in the generated dataset. For highly overlapping classes (large σ^2), the robust-max likelihood yields poor calibration and bad log-likelihood values.

than the softmax and Heaviside model. It is the only scalable approach that leads to well calibrated confidences and, hence, the *logistic-softmax* can be used as an efficient replacement of the standard softmax.

4.4.2 Effect of the augmentation

We investigate the effect of the augmentation of the *logistic-softmax* model and its variational approximation. To this end, we compare three different inference methods (1) variational inference for our augmented model (*Augmented VI*), (2) variational inference without augmentation (approximating the posterior of the original model directly from Section 4.2.1 using a variational Gaussian), where the gradients are computed via sampling (*VI*) and (3) Gibbs sampling (*Gibbs*), cf. Section 2.4. After burn-in, the samples from the Gibbs sampler serve as the ground truth since they come from the exact posterior. In this experiment, we do not use the inducing point approximation and all hyperparameters are fixed. We apply all three methods on

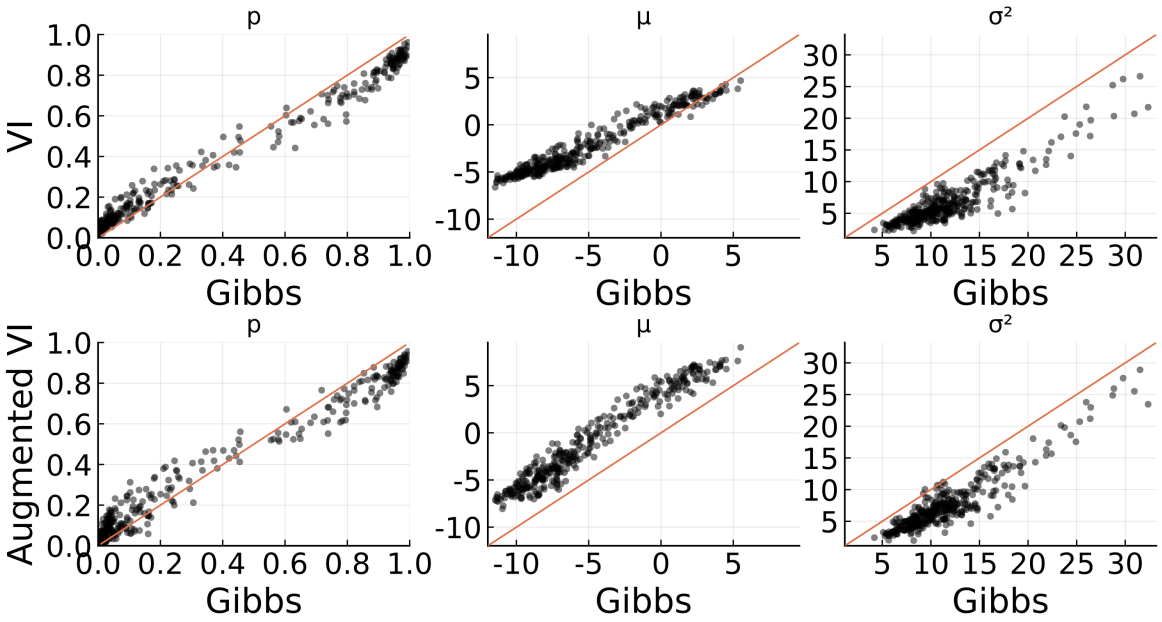

Figure 4.6: *Effect of the augmentation:* Comparison of the predictive marginals (p), posterior mean (μ) and posterior variance (σ^2) on a test set. Each plot shows the ground truth of the Gibbs sampler on the x-axis. On the y-axis the estimated values by variational inference without augmentation *VI* (top) and augmented variational inference *Augmented VI* are shown (bottom). Our efficient augmented VI method produces values very close to the less efficient VI method. Both methods slightly overestimate the mean (μ) and underestimate the variance (σ^2). However, for both methods, the final predictions (p) are close to the ground truth.

the dataset *Wine* (3 classes) and compare the predictive likelihood (p), and the mean (μ) and variance (σ^2) of the latent GPs on a test set. We compare each entry of the three-dimensional vectors p, μ, σ^2 with the ground truth and display the results for all classes $c = 1, 2, 3$ combined in Fig. 4.6.

Variational inference in the augmented model results in an approximate posterior that is very close to the variational inference solution in the original model. Both methods lead to a similar slight approximation error of the posterior mean μ and variance σ^2 and give predictive marginals p close to the ground truth. The Gibbs sampling approach has a final prediction accuracy of 0.98, whereby both variational inference methods have a final accuracy of 0.96. We find that the augmentation approach can be used as a scalable alternative to standard variational inference.

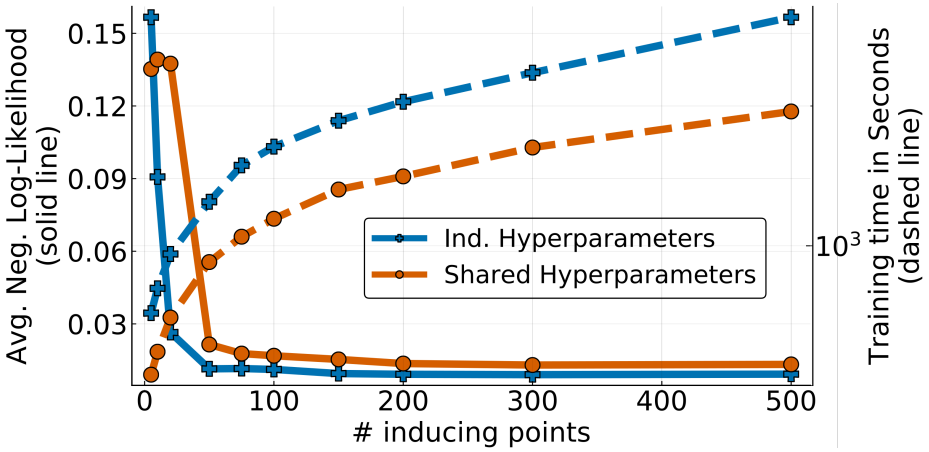

Figure 4.7: *Inducing points and hyperparameters:* The trade-off between predictive performance and run time is shown. Two versions of our method are used: individual hyperparameters for each GP (blue) and shared hyperparameters (orange). On the left y-axis we plot the negative log-likelihood (solid line) and on the right y-axis the training time (dashed line) as a function of the number of inducing points.

4.4.3 Inducing points and hyperparameters

In this experiment, we answer two questions. What is the effect of the number of inducing points and what is the difference between using shared hyperparameters and individual hyperparameters for each latent GP? We train our model on the *Shuttle* dataset (58,000 points, 9 classes) for 200 epochs. We vary the number of inducing points from 5 to 400, and set the GP hyperparameters to be either shared or independent among classes.

In Fig. 4.7 we display the trade-off between predictive performance and training time. We plot the negative log-likelihood (solid lines, y-axis left) and training time (dashed lines, y-axis right) as a function of the number of inducing points. If the number of inducing points is increased, the negative log-likelihood goes down and, oppositely, the training time goes up. We find that using only 200 inducing points already leads to near-optimal predictive performance. Using independent hyperparameters over shared hyperparameters does not lead to a significant improvement of the predictive performance but implies a higher computational cost, especially for datasets with a large number of classes.

4.4.4 Numerical comparison

Finally, we evaluate the predictive performance and convergence speed of our method against other state-of-the-art multi-class GP classification approaches. We compare

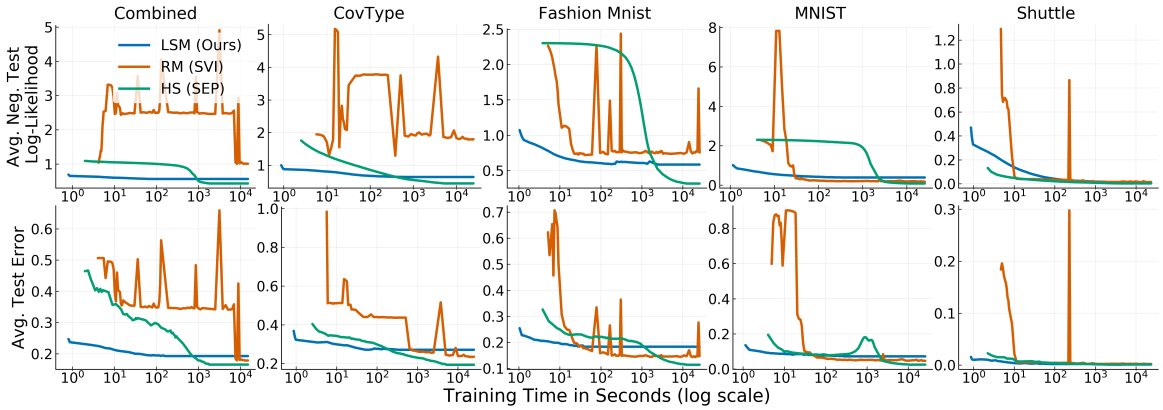

Figure 4.8: *Numerical comparison:* Prediction error and negative log-likelihood as a function of training time (seconds on a $\log_{10}$ scale). Our method (LSM) converges one to two orders of magnitude faster than the Heaviside model (HS) and is around 10 times faster than the robust-max model (RM). RM yields poor negative log-likelihood values due to poor uncertainty calibration.

our *logistic-softmax* likelihood model (LSM) trained via *augmented variational inference* against two competitors. First, the robust-max likelihood model (RM) by Hensman and Matthews (2015), which is provided in the package GPFlow (De G. Matthews et al., 2017) and trained by the natural gradient method of Salimbeni, Eleftheriadis, and Hensman (2018) and second, the Heaviside likelihood model (HS) trained by a scalable EP method (Villacampa-Calvo and Hernández-Lobato, 2017). For all methods, the hyperparameters are initialized to the same values and are optimized using Adam. We compare the methods on five different multi-class benchmark datasets: *Combined* (98,528 points, 50 features, 3 classes), *CovType* (581,000 points, 54 features, 7 classes), *Fashion-MNIST* (70,000 points, 784 features, 10 classes), *MNIST* (70,000 points, 784 features, 10 classes) and *Shuttle* (58,000 points, 9 features, 7 classes).

In Fig. 4.8, we plot the test error and negative log-likelihood as functions of the training time for each dataset. We find that our method (LSM) is one to two orders of magnitude faster than the EP based method for the Heaviside model (HS) and around ten times faster than the SVI based method for the robust-max model (RM).

Furthermore, our method consistently beats RM in terms of negative log-likelihood due to the better calibrated uncertainty quantification. Only on the *MNIST* dataset RM reaches a slightly better log-likelihood. This dataset is easily separable and therefore, suits well to the robust-max likelihood assumptions. On most datasets, the EP based method (HS) leads to slightly better predictive log-likelihood values but is demanding a much longer training time. In contrast to the log-likelihood, the pure

prediction error is not very sensitive to uncertainty calibration. All three methods achieve similar prediction errors whereby HS is a bit better on some datasets.

Moreover, the optimization curves in Fig. 4.8 show that our inference method is much more stable than the SVI approach for the RM model. This is due to our efficient coordinate ascent updates, which are given in closed form. The RM approach suffers from additional noise injected by approximating its gradients.

To summarize, our method is a good choice for fast inference on big datasets. It is particularly well fitted for datasets with overlapping classes where well calibrated uncertainty quantification is important. Due to the closed-form updates our method is more stable than the competitors.

Chapter 5

Bayesian Support Vector Machine

In this chapter, we propose a fast inference method for Bayesian nonlinear support vector machines that leverages stochastic variational inference and inducing points. Our experiments show that the proposed method is faster than competing Bayesian approaches and scales easily to millions of data points. It provides additional features over frequentist competitors such as accurate predictive uncertainty estimates and automatic hyperparameter search. The code is available via Github[1].

This chapter is based on:

> F. Wenzel, T. Galy-Fajou, M. Deutsch, M. Kloft (2017). "Bayesian Nonlinear Support Vector Machines for Big Data". In: *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases.*

The Support vector machine (SVM) (Cortes and Vapnik, 1995) is a classic supervised classification algorithm that separates the data points by a clear gap that is as wide as possible (maximal margin). The classic SVM does not exhibit a probabilistic formulation and therefore lacks a representation of uncertainty[2].

Recently, it was shown that the SVM admits a Bayesian interpretation through the technique of data augmentation (Polson and Scott, 2011; Henao, Yuan, and Carin, 2014). The so-called *Bayesian nonlinear SVM* combines the best of both worlds: it inherits the geometric interpretation, its robustness against outliers, state-of-the-art accuracy (Fernández-Delgado et al., 2014), and theoretical error guarantees (Mohri, Rostamizadeh, and Talwalkar, 2012) from the frequentist formulation of the SVM,

[1] `https://github.com/theogf/BayesianSVM`

[2] Note that frequentist approaches can also lead to other forms of uncertainty estimates by using post-hoc methods as e.g. Platt scaling. But since the classic SVM does not exhibit a probabilistic formulation these uncertainty estimates cannot be directly computed from the model.

but like Bayesian methods it also allows for flexible feature modeling, automatic hyperparameter tuning and predictive uncertainty quantification.

However, existing inference methods for the Bayesian support vector machine (such as the expectation conditional maximization method proposed by Henao, Yuan, and Carin, 2014) scale rather poorly with the number of samples and are limited in application to datasets with only thousands of data points. Based on stochastic variational inference (Hoffman et al., 2013) and inducing points (Hensman, Fusi, and Lawrence, 2013), we develop a *fast* and *scalable* inference method for the nonlinear Bayesian SVM. Our experiments demonstrate that our inference method can compete with corresponding frequentist approaches to SMVs in terms of scalability to big data, yet they offer additional benefits such as uncertainty estimation and automated hyperparameter search.

The Bayesian SVM model. Let $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$ be n observations where $\boldsymbol{x}_i \in \mathbb{R}^d$ is a feature vector with corresponding labels $y_i \in \{-1, 1\}$. The SVM aims to find an optimal score function f by solving the following regularized risk minimization objective:

$$\arg\min_f \ \gamma R\left(f\right) + \sum_{i=1}^{n} \max\left(0, 1 - y_i f(\boldsymbol{x}_i)\right), \tag{5.1}$$

where R is a regularizer function controlling the complexity of the decision function f, and γ is a hyperparameter to adjust the trade-off between training error and the complexity of f. The loss $\max\left(0, 1 - yf(\boldsymbol{x})\right)$ is called hinge loss. The classifier is then defined as $\text{sign}(f(\boldsymbol{x}))$.

A Bayesian formulation of the SVM is obtained by introducing a pseudo-likelihood function, which is proportional to the exponential hinge loss,

$$L\left(y_i|\boldsymbol{x}_i, f\right) = \exp\left(-2\max(1 - y_i f(\boldsymbol{x}_i), 0)\right). \tag{5.2}$$

It is a pseudo-likelihood since it is normalized in y (Polson and Scott, 2011), but the subsequent data augmentation approach will solve the issue.

For the case of a linear decision function, i.e. $f(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\beta}$, the SVM optimization problem (5.1) is equivalent to estimating the mode of the pseudo-posterior

$$p(\boldsymbol{\beta}|\mathcal{D}) \propto \prod_{i=1}^{n} L(y_i|\boldsymbol{x}_i^\top\boldsymbol{\beta})p(\boldsymbol{\beta}), \tag{5.3}$$

where $p(\boldsymbol{\beta})$ denotes a prior such that $\log p(\boldsymbol{\beta}) \propto -2\gamma R(\boldsymbol{\beta})$. In the following, we use the prior $\boldsymbol{\beta} \sim \mathcal{N}(0, S)$, where $S \in \mathbb{R}^{d\times d}$ is a positive definite matrix. From

a frequentist SVM view, this choice generalizes the usual L^2-regularization to non-isotropic regularizers. Note that our proposed framework can be easily extended to other regularization techniques by adjusting the prior on $\boldsymbol{\beta}$ (e.g. block $\ell_{(2,p)}$-norm regularization, which is known as multiple kernel learning (Kloft et al., 2011)).

Henao, Yuan, and Carin, 2014 develop a nonlinear (kernelized) version of this model. They assume a continuous decision function $f(\boldsymbol{x})$ to be drawn from a zero-mean Gaussian process $f \sim \mathrm{GP}(0, k)$, where k is a kernel function. This leads to the nonlinear Bayesian SVM posterior

$$p(f|\mathcal{D}) \propto \prod_{i=1}^{n} L(y_i|\boldsymbol{x}_i, f)p(f). \tag{5.4}$$

In Section 5.2, we discuss suitable augmentations for both models.

5.1 Background and related work

There has recently been significant interest in utilizing max-margin based discriminative Bayesian models for various applications. For example, Zhu et al., 2014 employ a max-margin based Bayesian classification to discover latent semantic structures for topic models, Xu, Zhu, and Zhang, 2013 use a max-margin approach for efficient Bayesian matrix factorization, and Zhang, Jun, and Zhang, 2014 develop a new max-margin approach to hidden Markov models.

All these approaches apply the Bayesian reformulation of the classic SVM introduced by Polson and Scott, 2011. This model is extended by Henao, Yuan, and Carin, 2014 to the nonlinear case. The authors show improved accuracy compared to standard methods such as (non-Bayesian) SVMs and Gaussian process (GP) classification.

However, the inference methods proposed by Polson and Scott, 2011 and Henao, Yuan, and Carin, 2014 have the drawback that they partially rely on point estimates of the latent variables and do not scale well to large datasets. Luts and Ormerod, 2014 apply mean field variational inference to the linear case of the model, but their proposed technique does not lead to substantial performance improvements and neglects the nonlinear model.

Uncertainty estimation for SVMs is usually done via Platt's technique (Platt, 1999), which consists of applying logistic regression on the function scores produced by the SVM. In contrast, our technique directly yields a sound predictive distribution instead of using a heuristically motivated transformation.

5.2 The augmentation

In this section, we introduce suitable augmentations for the linear Bayesian SVM and the nonlinear Bayesian SVM model.

The augmented linear Bayesian SVM. We first consider the linear version of the Bayesian SVM (5.3). By making use of integral identities stemming from function theory, Polson and Scott, 2011 show that the linear pseudo-likelihood L (5.2) can be expressed as

$$L(y_i|\boldsymbol{x}_i, \boldsymbol{\beta}) = \int_0^\infty \frac{1}{\sqrt{2\pi\lambda_i}} \exp\left(-\frac{1}{2}\frac{\left(1+\lambda_i - y_i\boldsymbol{x}_i^T\boldsymbol{\beta}\right)^2}{\lambda_i}\right) \mathrm{d}\lambda_i. \tag{5.5}$$

Building on this identity, we augment the model with a set of auxiliary variables $\boldsymbol{\lambda} := (\lambda_1, \ldots, \lambda_n)^\top$ leading to the augmented joint distribution given by

$$\begin{aligned} p(y_i|\boldsymbol{x}_i, \boldsymbol{\beta}, \lambda_i) &= \frac{1}{\sqrt{2\pi\lambda_i}} \exp\left(-\frac{1}{2}\frac{\left(1+\lambda_i - y_i\boldsymbol{x}_i^T\boldsymbol{\beta}\right)^2}{\lambda_i}\right) \\ p(\lambda_i) &= \mathbb{1}_{[0,\infty)}(\lambda_i), \end{aligned}$$

where we impose an improper prior on λ_i. The improper prior is unproblematic since the complete conditional distributions are proper again. They are given in closed form,

$$\begin{aligned} \boldsymbol{\beta}|\boldsymbol{\lambda}, \Sigma, \mathcal{D} &\sim \mathcal{N}\left(B(\lambda^{-1}+1),\, B\right), \\ \lambda_i|\boldsymbol{\beta}, \mathcal{D}_i &\sim \mathrm{GIG}\left(1/2, 1, (1-y_i\boldsymbol{x}_i^\top\boldsymbol{\beta})^2\right), \end{aligned} \tag{5.6}$$

where $X = (\boldsymbol{x}_1, ..., \boldsymbol{x}_n)^\top \in \mathbb{R}^{n\times d}$, $Y = \mathrm{diag}(\boldsymbol{y})$, $Z = YX$, $B^{-1} = Z\Lambda^{-1}Z^\top + S^{-1}$, $\Lambda = \mathrm{diag}(\boldsymbol{\lambda})$ and where GIG denotes the generalized inverse Gaussian distribution. The n latent variables λ_i of the model scale the variance of the full posteriors locally. The model thus constitutes a special case of a normal variance-mean mixture,

The augmented nonlinear Bayesian SVM. Henao, Yuan, and Carin, 2014 propose an augmentation for the nonlinear Bayesian SVM by substituting the linear function $\boldsymbol{x}_i^\top\boldsymbol{\beta}$ by $f_i := f(\boldsymbol{x}_i)$ in (5.5) and obtain the conditional posteriors

$$\begin{aligned} \boldsymbol{f}|\boldsymbol{\lambda}, \mathcal{D} &\sim \mathcal{N}\left(CY(\boldsymbol{\lambda}^{-1}+1),\, C\right), \\ \lambda_i|f_i, \mathcal{D}_i &\sim \mathrm{GIG}\left(1/2, 1, (1-y_i f_i)^2\right), \end{aligned} \tag{5.7}$$

with $C^{-1} = \Lambda^{-1} + K^{-1}$. For a test point $\boldsymbol{x}_*$ the conditional predictive distribution for $f_* = f(\boldsymbol{x}_*)$ under this model is

$$f_* | \boldsymbol{\lambda}, \boldsymbol{x}_*, \mathcal{D} \sim \mathcal{N}\left(\boldsymbol{k}_*^\top (K + \Lambda)^{-1} Y (1 + \boldsymbol{\lambda}),\, k_{**} - \boldsymbol{k}_*^\top (K + \Lambda)^{-1} \boldsymbol{k}_*\right),$$

where $K := k(X, X)$, $\boldsymbol{k}_* := k(X, \boldsymbol{x}_*)$, $k_{**} := k(\boldsymbol{x}_*, \boldsymbol{x}_*)$. The conditional class membership probability is

$$p(y_* = 1 | \boldsymbol{\lambda}, \boldsymbol{x}_*, \mathcal{D}) = \Phi\left(\frac{\boldsymbol{k}_*^T (K + \Lambda)^{-1} Y (1 + \boldsymbol{\lambda})}{1 + k_{**} - \boldsymbol{k}_*^\top (K + \Lambda)^{-1} \boldsymbol{k}_*}\right),$$

where $\Phi(.)$ denotes the probit link function.

Note that the conditional posteriors as well as the class membership probability still depend on the local latent variables λ_i. We are interested in the marginal predictive distributions, but unfortunately, the latent variables cannot be integrated out analytically. Both Polson and Scott, 2011 and Henao, Yuan, and Carin, 2014 propose MCMC algorithms and stepwise inference schemes similar to an EM algorithm to approach this problem. However, these methods do not scale well to big data problems and the probability estimation still relies on point estimates of the n-dimensional $\boldsymbol{\lambda}$. We overcome these problems by proposing a scalable inference method and obtain an approximate marginal predictive distribution (where $\boldsymbol{\lambda}$ is integrated out instead of being point estimated).

5.3 Inference

We approximate the latent GP $\boldsymbol{f}$ by a *sparse GP* building on *inducing points*. As outlined in Section 1.2, we introduce M inducing points $\boldsymbol{u}$ and connect the GP values with the inducing points via the joint prior distribution $p(\boldsymbol{f}, \boldsymbol{u}) = p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})$ given in Eq. 1.5. We apply a structure mean-field approach (Wainwright and Jordan, 2008) to the augmented model

$$p(\boldsymbol{y}, \boldsymbol{u}, \boldsymbol{f}, \boldsymbol{\lambda}) = p(\boldsymbol{y}, \boldsymbol{\lambda}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{u})p(\boldsymbol{u})$$

leading to an approximate posterior of the sparse GP $\boldsymbol{u}$ and the augmented auxiliary variables $\boldsymbol{\lambda}$

$$p(\boldsymbol{u}, \boldsymbol{\lambda}|\boldsymbol{y}) \approx q(\boldsymbol{u})q(\boldsymbol{\lambda}).$$

Setting the functional derivative of the ELBO zero gives the family of variational distributions

$$q(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \Sigma)$$
$$q(\lambda_i) = \mathrm{GIG}(\frac{1}{2}, 1, \alpha_i),$$

where the optimal variational distribution $q(\lambda_i)$ is in a restricted distribution class with only one free variational parameter α_i. The ELBO is derived in closed form and presented in Appendix C.1.

The natural gradients of $\mathcal{L}$ w.r.t. the Gaussian natural parameters $\boldsymbol{\eta}_1 = \zeta^{-1}\boldsymbol{\mu}$, $\eta_2 = -\frac{1}{2}\zeta^{-1}$ are

$$\widetilde{\nabla}_{\boldsymbol{\eta}_1}\mathcal{L} = \boldsymbol{\kappa}^\top Y(\alpha^{-\frac{1}{2}} + 1) - \boldsymbol{\eta}_1 \tag{5.8}$$
$$\widetilde{\nabla}_{\eta_2}\mathcal{L} = -\frac{1}{2}(K_{mm}^{-1} + \kappa^\top A^{-\frac{1}{2}}\kappa) - \eta_2, \tag{5.9}$$

with $A = \mathrm{diag}(\alpha)$. The natural gradient updates always lead to a positive definite covariance matrix[3] and in our implementation ζ has not to be parametrized in any way to ensure positive-definiteness. The derivative of $\mathcal{L}$ w.r.t. α_i is

$$\nabla_\alpha\mathcal{L} = \frac{(1 - y_i\boldsymbol{\kappa}_i\boldsymbol{\mu})^2 + y_i(\boldsymbol{\kappa}_i\zeta\boldsymbol{\kappa}_i^\top + \widetilde{K}_{ii})y_i}{4\sqrt{\alpha_i}^3} - \frac{1}{4\sqrt{\alpha_i}}, \tag{5.10}$$

where $\kappa = K_{nm}K_{mm}^{-1}$. Setting it zero gives the coordinate ascent update for α_i,

$$\alpha_i = (1 - y_i\boldsymbol{\kappa}_i\boldsymbol{\mu})^2 + y_i(\boldsymbol{\kappa}_i\zeta\boldsymbol{\kappa}_i^\top + \widetilde{K}_{ii})y_i.$$

Details can be found in Appendix C.2.

The inducing point locations can be either treated as hyperparameters and optimized while training (Titsias, 2009) or can be fixed before optimizing the variational objective. In our experiments, we have observed that selecting the inducing locations by the k-means clustering algorithm (kMeans) (Murphy, 2012) and fixing them while training yields the best results. We apply the adaptive learning rate method described in Ranganath et al., 2013 and present our algorithm in Alg. 2.

Uncertainty predictions. Besides the advantage of automated hyperparameter tuning, the probabilistic formulation of the SVM leads directly to uncertainty estimates of the predictions. The standard SVM lacks this capability, and only heuristic approaches as e.g. Platt scaling (Platt, 1999) exist. Using the approximate posterior

[3]This follows directly since K_{mm} and $A^{-\frac{1}{2}}$ are positive definite.

Algorithm 2 Augmented VI for the Bayesian nonlinear SVM

set the learning rate schedule ρ_t appropriately
initialize η_1, η_2
select m inducing points locations (e.g. via kMeans)
compute kernel matrices K_{mm}^{-1} and $\widetilde{K} = K_{nn} - K_{nm}K_{mm}^{-1}K_{mn}$
while not converged **do**
    get $\mathcal{S}$ = mini-batch index set of size s
    update $\alpha_i = (1 - y_i\kappa_i\mu)^2 + y_i(\kappa_i\zeta\kappa_i^\top + \widetilde{K}_{ii})y_i$
    compute $A_\mathcal{S} = \mathrm{diag}(\alpha_i, \;\; i \in \mathcal{S})$
    compute $\hat{\eta}_1 = \kappa^\top Y(\alpha^{-\frac{1}{2}} + 1)$
    compute $\hat{\eta}_2 = -\frac{1}{2}(K_{mm}^{-1} + \kappa^\top A^{-\frac{1}{2}}\kappa)$
    update $\eta_1 = (1 - \rho_t)\eta_1 + \rho_t\hat{\eta}_1$
    update $\eta_2 = (1 - \rho_t)\eta_2 + \rho_t\hat{\eta}_2$
    compute $\zeta = -\frac{1}{2}\eta_2^{-1}$
    compute $\mu = \zeta\eta_1$
return $\alpha_1, \ldots, \alpha_n, \mu, \zeta$

$q(\boldsymbol{u}|\mathcal{D}) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \zeta)$ obtained by our stochastic variational inference method (Alg. 2) we compute the class membership probability for a test point $\boldsymbol{x}^*$,

$$\begin{aligned} p(f^*|\boldsymbol{x}^*, \mathcal{D}) &= \int p(y^*|\boldsymbol{u}, \boldsymbol{x}^*)p(\boldsymbol{u}|\mathcal{D})\mathrm{d}\boldsymbol{u} \\ &\approx \int p(y^*|\boldsymbol{u}, \boldsymbol{x}^*)q(\boldsymbol{u}|\mathcal{D})\mathrm{d}\boldsymbol{u} \\ &= \mathcal{N}\left(y^*|\boldsymbol{K}_{*m}K_{mm}^{-1}\boldsymbol{\mu},\; K_{**} - \boldsymbol{K}_{*m}K_{mm}^{-1}(\boldsymbol{K}_{m*} + \zeta K_{mm}^{-1}\boldsymbol{K}_{m*})\right) \\ &=: q(f^*|\boldsymbol{x}^*, \mathcal{D}), \end{aligned}$$

where $\boldsymbol{K}_{*m}$ denotes the kernel matrix between test and inducing points and K_{**} the kernel matrix between test points. This leads to the approximate class membership distribution

$$q(y^*|\boldsymbol{x}^*, \mathcal{D}) = \Phi\left(\frac{\boldsymbol{K}_{*m}K_{mm}^{-1}\boldsymbol{\mu}}{K_{**} - \boldsymbol{K}_{*m}K_{mm}^{-1}(\boldsymbol{K}_{m*} + \zeta K_{mm}^{-1}\boldsymbol{K}_{m*}) + 1}\right) \tag{5.11}$$

where $\Phi(.)$ is the probit link function. Note that we already computed inverse K_{mm}^{-1} for the training procedure leading to a computational overhead stemming only from simple matrix multiplication. Our experiments show that (5.11) leads to reasonable uncertainty estimates.

Optimization of hyperparameters. The probabilistic formulation of the SVM lets us directly learn the hyperparameters while training. To this end we maximize

the marginal likelihood $p(\boldsymbol{y}|h)$, where h denotes the set of hyperparameters. We follow an approximate approach as explained in Section 1.2.

The standard SVM does not exhibit a probabilistic formulation and the hyperparameters have to be tuned via computationally expensive methods as e.g. grid search and cross-validation. Our approach allows to estimate the hyperparameters during training time and follows gradients instead of only testing single hyperparameters.

In Appendix C.3, we provide the gradient of the variational objective $\mathcal{L}$ w.r.t. to a general kernel and show how to optimize arbitrary differentiable hyperparameters. Our experiments exemplify our automated hyperparameter tuning approach by optimizing the hyperparameters of a squared exponential kernel (cf. Section 1.2).

5.4 Experiments

We compare our stochastic Bayesian SVM approach (S-BSVM) against the expectation conditional maximization (ECM) method for the Bayesian nonlinear SVM proposed by Henao, Yuan, and Carin (2014). We furthermore compare against standard GP classification (GPC) using expectation propagation, and the standard (non-probabilistic) SVM together with Platt scaling (Chang and Lin, 2011; Platt, 1999) (SVM + PLATT) using the package libSVM (Chang and Lin, 2013). Note that there also exist alternative scalable inference methods for GP classification, as e.g. the method presented in Chapter 3. We only include one GP classification inference method since the focus of the experiments is on comparing methods that are related to the SVM. For all experiments, we employ a squared exponential kernel (cf. Section 1.2) with length-scale parameter θ. We perform all experiments using only one CPU core with 2.9 GHz and 386 GB RAM.

5.4.1 Prediction performance and uncertainty estimation

We experiment on seven real-world datasets and compare the prediction performance, the quality of the uncertainty estimates and run time of the methods. The results are presented in Table 5.1. We show that our method (S-BSVM) is up to 22 times faster than the direct competitor ECM and up to 700 times faster than GP classification while outperforming the competitors in terms of prediction performance and quality of uncertainty estimates in most cases. The non-probabilistic SVM is naturally the fastest method. Using the heuristic Platt scaling approach, it leads to heuristic class membership probabilities but it still lacks the advantages of a real probabilistic

Dataset	n	dim.		**S-BSVM**	ECM	GPC	SVM + Platt
Breast Cancer	263	9	Error	**.26 ± .07**	.27 ± .10	.27 ± .07	.27 ± .09
			Brier Score	**.18 ± .03**	.19 ± .05	**.18 ± .03**	.19 ± .04
			Time [s]	0.32	1.4	6.7	**0.04**
Diabetes	768	8	Error	**.22 ± .06**	.25 ± .07	.23 ± .07	.24 ± .07
			Brier Score	.16 ± .04	.17 ± .04	**.15 ± .04**	.16 ± .04
			Time [s]	3.9	33	67	**0.11**
Flare	144	9	Error	**.36 ± .12**	**.36 ± .12**	**.36 ± .11**	**.36 ± .12**
			Brier Score	**.22 ± .05**	.25 ± .07	.24 ± .03	.24 ± .04
			Time [s]	0.08	0.26	1.8	**0.01**
German	1000	20	Error	**.24 ± .11**	.25 ± .12	.25 ± .13	.27 ± .10
			Brier Score	**.17 ± .06**	**.17 ± .05**	**.17 ± .06**	.18 ± .05
			Time [s]	12	80	115	**0.15**
Heart	270	13	Error	**.16 ± .06**	.19 ± .09	**.16 ± .06**	.17 ± .07
			Brier Score	.13 ± .04	.14 ± .04	**.12 ± .03**	**.12 ± .04**
			Time [s]	0.34	2.2	6	**0.04**
Splice	2991	60	Error	.13 ± .03	**.11 ± .03**	.32 ± .14	.14 ± .01
			Brier Score	.17 ± .01	.18 ± .01	.40 ± .14	**.11 ± .01**
			Time [s]	18	406	419	**1.3**
Waveform	5000	21	Error	**.09 ± .02**	.10 ± .02	.10 ± .02	.10 ± .02
			Brier Score	**.06 ± .01**	.15 ± .01	**.06 ± .01**	**.06 ± .01**
			Time [s]	12.5	264	8691	**2.3**

Table 5.1: Average prediction error and Brier score with one standard deviation, and computation time in seconds are shown.

model (as e.g. uncertainty quantification of the learned parameters and automatic hyperparameter tuning).

To evaluate the quality of the uncertainty estimates we compute the Brier score, which is considered as a good performance measure for probabilistic predictions (Brier, 1950) and is defined as $BS = \frac{1}{n}\sum_{i=1}^{N}\left(y_i - p(y_i|\boldsymbol{x}_i)\right)^2$, where $y_i \in \{0,1\}$ is the observed output and $p(y_i|\boldsymbol{x}_i) \in [0,1]$ is the predicted class membership probability. Note that smaller Brier score indicates better performance.

The datasets are all from the Rätsch benchmark datasets (Diethe, 2015) and the methods are evaluated using a 10-fold cross-validation. For S-BSVM we choose the number of inducing points as 20% of the training set size, except for the datasets *Splice*, *German* and *Waveform*, where we use 100 inducing points. For each dataset mini-batches of 10 samples are used.

5.4.2 Big data experiment

In the previous experiments, we considered rather small datasets since the direct competitor ECM does not scale to big datasets. In this experiment, we demonstrate the

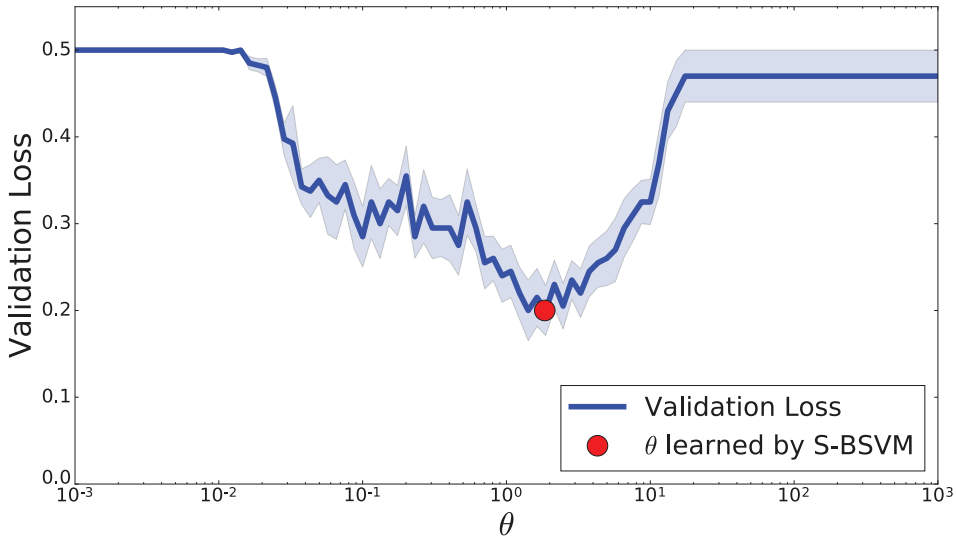

Figure 5.1: The average validation loss as a function of the length-scale parameter θ of a squared exponential kernel, computed by grid search and 10-fold cross-validation, is shown. Our proposed automatic tuning approach finds the true hyperparameter. The selected hyperparameter is indicated by the red circle.

superior scalability of our method. We consider the SUSY dataset (Baldi, Sadowski, and Whiteson, 2014) containing 5 million points with 17 features. We use a squared exponential kernel[4], 64 inducing points and mini-batches of 100 points. The training of our model takes only 10 minutes without any parallelization.

We use the area under the receiver operating characteristic curve (AUC) as performance measure since it is a standard evaluation measure on this dataset (Baldi, Sadowski, and Whiteson, 2014). Our method achieves an AUC of 0.84 and a Brier score of 0.22, whereby the state-of-the-art obtains an AUC of 0.88 using a deep neural network (5 layers, 300 hidden units each) (Baldi, Sadowski, and Whiteson, 2014). Note that this approach takes much longer to train and does not include uncertainty estimates.

5.4.3 Auto tuning of hyperparameters

The hyperparameters of the standard SMV are usually selected via grid search, which is often slow. As we have shown, our inference method possesses the ability of automatic hyperparameter tuning. In this experiment, we demonstrate that our method, indeed, finds the optimal length-scale hyperparameter of a squared exponential ker-

[4]The length scale parameter tuning is not included in the training time. We found $\theta = 5.0$ by our proposed automatic tuning approach.

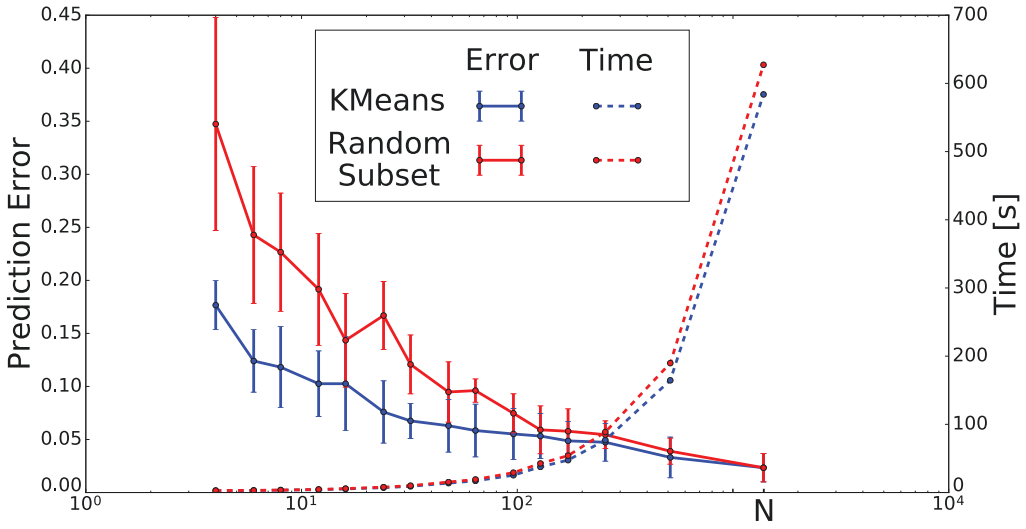

Figure 5.2: Average prediction error and training time as functions of the number of inducing points selected by two different methods with one standard deviation (using 10-fold cross-validation).

nel. We apply the hyperparameter optimizing scheme outlined in Section 5.3 and alternate between ten variational parameter updates and one hyperparameter update. We compute the true validation loss of the length-scale parameter θ by a grid search approach that consists of training our model (S-BSVM) for each θ and measuring the prediction performance using 10-fold cross-validation. In Fig. 5.1 we plot the validation loss and the length-scale parameter found by our method. We find the true optimal value by only using 5 hyperparameter optimization steps. Training and hyperparameter optimization takes only 0.3 seconds for our method, whereas grid search takes 188 seconds (with a grid size of 1000 points).

5.4.4 Inducing points selection

The sparse GP model used in our inference scheme builds on a set of inducing points where both the number and the locations of the inducing points are free parameters. We investigate three different inducing point selection methods: random subset selection from the training set, the Gaussian Mixture Model (GMM), and the k-means clustering algorithm with an improved k-means++ seeding (kMeans) (Bachem et al., 2016). Furthermore, we show how the number of inducing points affects the prediction accuracy and the run time. In Fig. 5.2, we present the results for the USPS dataset (Lichman, 2013), which we reduced to a binary problem using only the digits 3 and 5 (N=1350 and d=256). We observe similar results on all datasets we have

considered. For all methods we progressively increase the number of inducing points and compute the prediction error by 10-fold cross-validation.

The GMM is unable to fit large numbers of samples and dimensions and fails to converge for all datasets tried, therefore, we do not include it in the plot. For small numbers of inducing points, the k-means selection algorithm leads to much better prediction performance than random subset selection. Furthermore, we show that using only a small fraction of inducing points (around 1% of the original dataset) leads to a nearly optimal prediction performance by simultaneously significantly decreasing the run time.

Chapter 6

Sparse Gaussian Process Linear Mixed Model: Feature Extraction in Confounded Data

In this chapter, we propose a new model for sparse feature selection in a binary classification setting where the training data show spurious correlations, e.g., due to confounding. An important application is to find a sparse set of genetic traits that best predict a binary phenotype of interest, while simultaneously correcting for various confounding factors such as age, ethnicity and population structure. Our new model, the *sparse Gaussian process linear mixed model* (Sparse GP-LMM), generalizes the LMM-Lasso, which is restricted to continuous outcomes (regression), to binary outcomes (classification). As a technical challenge, the model no longer possesses a closed-form likelihood function. We present two scalable approximate inference algorithms for two versions of our model; an expectation propagation based approach for the Sparse GP-LMM with a *probit likelihood* and *augmented variational inference* method for the Sparse GP-LMM with a *logistic likelihood*. The EP method leads to the best prediction performance but is limited to rather small datasets containing a few hundred data points, whereas the *augmented variational inference* method scales to datasets with millions of points by introducing a small additional approximation error. We show on three real-world examples from different domains that in the setup of binary labels, our algorithms lead to better prediction accuracies and also select features that show less correlation with the confounding factors. The code is available via Github[1].

[1]The EP based method for the Probit GP-LMM can be found at `https://github.com/flwenzel/Probit-LMM` and the augmented variational inference based method can be found at `https://gitlab.tubit.tu-berlin.de/lenz3000/Sparse-Probit-GP-LMM`.

This chapter is based on:

> S. Mandt*, F. Wenzel*, S. Nakajima, J. P. Cunningham, C. Lippert, M. Kloft (2017). "Sparse Probit Linear Mixed Model". *Machine Learning, 106(9), 1621-1642.*
>
> F. Wenzel, S. Mandt, M. Kloft (2019). "Scalable Feature Extraction in Confounded Data". In: *Proceedings of the Southern California Machine Learning Symposium.*

Genetic association studies have emerged as an important branch of statistical genetics (Manolio et al., 2009; Vattikuti et al., 2014). The goal of this field is to find causal associations between high-dimensional vectors of *genotypes*, such as single nucleotide polymorphisms (SNPs) and observable outcomes (*phenotypes, or traits*). For various phenotypes, such as heritable diseases, it is assumed that these associations manifest themselves on only a small number of genes. This leads to the challenging problem of identifying a few relevant positions along the genome among ten thousands of irrelevant genes. For various complex diseases, such as bipolar disorder or type 2 diabetes (Craddock, Hurles, Cardin, et al., 2010), these sparse associations are largely unknown (Manolio et al., 2009), which is why these missing associations have been entitled the *"The Dark Matter of Genomic Associations"*[2].

Genetic associations can be spurious, unreliable and unreproducible when the data are subject to spurious correlations due to confounding (Imbens and Rubin, 2015; Pearl et al., 2009; Morgan and Winship, 2014). Confounding can stem from varying experimental conditions and demographics such as age, ethnicity, or gender (Li, Rakitsch, and Borgwardt, 2011). The perhaps most important types of confounding in statistical genetics arise from population structure (Astle and Balding, 2009), as well as similarities between closely related samples (Li, Rakitsch, and Borgwardt, 2011; Lippert et al., 2011; Fusi, Stegle, and Lawrence, 2012). Ignoring such confounders can often lead to spurious false-positive findings that cannot be replicated on independent data (Kraft, Zeggini, and Ioannidis, 2009). Correcting for such confounding dependencies is considered one of the greatest challenges in statistical genetics (Vilhjálmsson and Nordborg, 2013).

*Equal contributions.

[2]See the proceedings of the "Workshop on the Dark Matter of Genomic Associations With Complex Diseases: Explaining the Unexplained Heritability From Genome-Wide Association Studies" published by The National Human Genome Research Institute 2009.

Confounding and similarity kernels. The problem of confounding is fundamental in statistics. A confounder is a common cause both of the genotypes and the traits. When it is unobserved, it induces spurious correlations that have no causal interpretation: we say that the genotypes and traits are *confounded* (Imbens and Rubin, 2015; Pearl et al., 2009; Morgan and Winship, 2014).

In statistical genetics, a major source of confounding originates from population structure (Astle and Balding, 2009). Population structure implies that due to common ancestry, individuals that are related co-inherit a large number of genes, making them more similar to each other, whereas individuals of unrelated ancestry obtain their genes independently, making them more dissimilar. For this reason, collecting genetic data has to be done carefully. For example, when data are collected only in selected geographical areas (such as in specific hospitals), one introduces a selection bias into the sample, which can induce spurious associations between phenotypes and common genes in the population. It is an active area of research to find models that are less prone to confounding (Vilhjálmsson and Nordborg, 2013). In this paper, we present such a model for the setup of binary classification.

A popular approach to correcting for confounding relies on modeling the confounding by a random effect based on a similarity kernel, also called kinship matrix (Astle and Balding, 2009). Given n samples, we can construct an $n \times n$ matrix K that quantifies the similarity between samples based on some arbitrary measure. In the case of confounding by population structure, a popular choice is the linear kernel function $K_{ij} = \boldsymbol{x}_i^\top \boldsymbol{x}_j$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ is a vector of genetic features of individual i. In our model (6.1), we use a generalized version of this approach by allowing K to be an arbitrary kernel matrix and the confounding effect is modeled by a GP. The kernel can also be based on additional side information $\tilde{\boldsymbol{x}}$ (which is not included in the feature vector $\boldsymbol{x}$). Details of constructing similarity kernels and other examples can be found in Astle and Balding, 2009. Next, we introduce our new model for feature extraction in confounded data with binary labels.

6.1 Sparse Gaussian process linear mixed model

Our approach is inspired by linear mixed models (LMMs) for genome-wide association studies (Lippert et al., 2011), which model the effects of confounding in terms of correlated noise on the traits. A related tool for feature selection is the LMM-Lasso (Rakitsch et al., 2013). In this paper, we extend the idea of LMMs to binary labels. The LMM and its Lasso version are restricted to the linear regression case

where the output variable is continuous, but in many important applications, the phenotype is binary, such as the presence or absence of a heritable disease. To this end, we threshold the output through an activation function mapping the outputs to the interval $[0, 1]$. This makes parameter learning challenging since the model becomes a latent GP model with an intractable likelihood.

We propose the *sparse Gaussian process linear mixed model* (sparse GP-LMM), which is defined as follows. Let $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^\top \in \mathbb{R}^{n \times d}$ be the data matrix of the d-dimensional training points with labels $\boldsymbol{y} = (y_1, \ldots, y_n) \in \{-1, 1\}^n$. The score for each data point is given by

$$s_i = \boldsymbol{\beta}^\top \boldsymbol{x}_i + f(\boldsymbol{x}_i),$$

which consists of two contributions. First, a linear term parameterized by a *sparse weight vector* $\boldsymbol{\beta}$, which models the true underlying fixed effect. We place a Laplace prior over the linear weights,

$$\boldsymbol{\beta} \sim \text{Laplace}(0, \lambda_0^{-1}).$$

This corresponds to the LASSO (Tibshirani, 1996) and results in a sparse weight vector, i.e. only a small number of important features are selected. The second contribution is $f(x_i)$, which models confounding by means of a *correlated noise term*. The more similar two data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are, the higher is the correlation of their noise contributions $f(\boldsymbol{x}_i)$, $f(\boldsymbol{x}_j)$. We place a Gaussian process prior over $f \sim \text{GP}(0, k)$, whereby the kernel function k encodes similarity with respect to a potential confounder. The crucial idea behind this approach is that parts of the score function that leads to the observed labels y_i can be explained away by the random effect, which is correlated with the confounder. The remaining part is governed by the sparse linear effect and is less correlated with the confounder and, therefore, is expected to lead to a better estimate of the true causal effect.

In order to obtain a likelihood suitable for classification, the score values are mapped to probabilities in the interval $[0, 1]$ via an activation function $\sigma(.)$ leading to a Bernoulli likelihood

$$p(y|\boldsymbol{f}, \boldsymbol{\beta}) = \prod_{i=1}^{n} \sigma(y_i s_i).$$

In our work, we consider two likelihoods, the probit likelihood function $\sigma(z) = \Phi(z)$, where $\Phi(.)$ is normal cumulative density function and the logistic likelihood function

$\sigma(z) = (1 + \exp(-z))^{-1}$. The joint distribution of the labels $\boldsymbol{y}$, the sparse weight vector $\boldsymbol{\beta}$ and the latent GP values $\boldsymbol{f}$ is

$$\begin{aligned} p(\boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{f}) &= p(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{f})p(\boldsymbol{f})p(\boldsymbol{\beta}) \\ &= \prod_{i=1}^{n} \sigma\left(y_i(\boldsymbol{\beta}^\top \boldsymbol{x}_i + f_i)\right) p(\boldsymbol{f})p(\boldsymbol{\beta}), \end{aligned} \tag{6.1}$$

where we use the abbreviation $f_i = f(\boldsymbol{x}_i)$.

6.2 Background and related work

In the following, we review related work. In particular, our model can be viewed as a generalization of the LMM-Lasso to binary outcomes and covers sparse probit regression and GP classification as limiting cases.

Linear mixed models. There is a large amount of literature on linear mixed models for genome-wide association studies. For a review see Price et al., 2010; Astle and Balding, 2009; Lippert, 2013. Our approach mostly relates to the the LMM-Lasso (Rakitsch et al., 2013). The LMM-Lasso can be recovered from our model (6.1) by using a Gaussian likelihood $p(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{f})$ instead of Bernoulli likelihood. Compared to feature selection in a simple linear regression model, the LMM-Lasso improves the selection of true non-zero effects as well as prediction quality (Rakitsch et al., 2013). Our model is a natural extension of this model to binary outcomes, such as the disease status of a patient. While one could also use the LMM-Lasso to model such binary labels, we show in our experimental section that this leads to lower predictive accuracies. Inference in our model is, however, more challenging than in the LMM-Lasso.

Limiting cases. Our model furthermore captures two limiting cases. First, we obtain *sparse probit regression* when using the *probit likelihood* and using a indentity matrix as kernel, $K = \lambda I$, where λ is a hyperparameter. In this case, the generative process for $y_i \sim \mathcal{B}(\boldsymbol{\beta}^\top \boldsymbol{x}_i + f(\boldsymbol{x}_i))$ simplifies to

$$\begin{aligned} y_i &= \operatorname{sign}(\boldsymbol{\beta}^\top \boldsymbol{x}_i + \epsilon_i) \\ \epsilon_i &\sim \mathcal{N}(0, \lambda + 1), \quad \boldsymbol{\beta} \sim \text{Laplace}(0, \lambda_0^{-1}) \end{aligned} \tag{6.2}$$

where $\mathcal{B}$ is the Bernoulli distribution. As an aside, we remark that using (a multiple of) an identity kernel $K = \lambda I$ together with a *logistic likelihood* function can be viewed

as a hybrid between probit regression and logistic regression since it is equivalent to the following latent noise model

$$y_i = \text{sign}(\boldsymbol{\beta}^\top \boldsymbol{x}_i + \epsilon_i + \tilde{\epsilon}_i)$$

with $\epsilon_i \sim \mathcal{N}(0, \lambda)$ and $\tilde{\epsilon}_i$ is a logistic noise variable $\tilde{\epsilon}_i \sim \text{Logistic}(0, 1)$.

Second, we obtain *GP classification*, by omitting the fixed effect (i.e., we set $\boldsymbol{\beta} = 0$). When properly trained, our model is thus expected to outperform both approaches in terms of prediction performance. We compare our method to the related methods LMM-Lasso, probit regression and GP classification in the experimental part of the paper and show enhanced accuracy.

Feature selection and confounding. There is a large body of work on feature selection in Lasso regression (Tibshirani, 1996). Alternative sparse priors to the Lasso have been suggested in Mohamed, Heller, and Ghahramani, 2011; Carbonetto, Stephens, et al., 2012. The joint problem of sparse estimation in a correlated noise setup has been restricted to the linear regression case (Seeger and Nickisch, 2011; Vattikuti et al., 2014; Rakitsch et al., 2013), whereas we are interested in classification. For classification, we remark that the ccSVM (Li, Rakitsch, and Borgwardt, 2011) deals with confounding in a different way and it does not yield a sparse solution. Finally, our algorithm builds on performing inference in a GP classification model, but note that standard GP classification does not yield a sparse estimate of features and therefore does not allow us to select predictive features.

Several alternatives to the LMM have recently been proposed and shall briefly be addressed. Song, Hao, and Storey (2015) develop a new statistical association test between traits and genetic markers. The approach reverses the placement of trait and genotype in the model and thus regresses the genotypes conditioned on the trait and an adjustment based on a fitted population structure model. Klasen et al. (2016) propose a new hierarchical testing procedure, where one searches for highly correlated clusters of genotypes and tests them for significant associations to the response variable. The significant clusters in the lowest hierarchy (or individual genotypes) are then considered as the causal genotypes of interest.

6.3 Inference

For each version of our model—the *probit GP linear mixed model*, which uses a probit likelihood and the *logistic GP linear mixed model*, which uses a logistic likelihood—we

develop an individual inference algorithm. For the probit likelihood model, we propose an inference method that is based on expectation propagation (EP) (Minka, 2001; Opper and Winther, 2001). This method computes an approximate MAP estimate of the linear effect $\boldsymbol{\beta}$ building on an approximate expectation maximization approach. For the logistic likelihood model, we propose a scalable variational inference (VI) method. This method can be used for two different inference tasks. It can be used to compute an approximate maximum a posteriori (MAP) estimate of $\boldsymbol{\beta}$. But it also allows for fully Bayesian inference by approximating the whole posterior distribution $p(\boldsymbol{\beta}|\boldsymbol{y})$ instead of computing only a single point estimate. The VI method builds on a probabilistic data augmentation approach and scales to big datasets.

Both inference methods are complementary to each other. They perform inference in two different versions of our model (EP for the probit likelihood model and VI for the logistic likelihood model). The VI approach lets us answer more general inference tasks since it can be used in the fully Bayesian setting, whereas the EP approach is limited to the MAP inference setting. The most important difference is that the VI method scales to datasets containing millions of data points, whereas the EP method is constrained on rather small datasets containing a few thousand points. On the other hand, in the small data regime, the EP method is expected to lead to better prediction performance since it is built on more accurate approximations. In the following, we present the VI and EP based inference methods.

6.4 Inference via expectation propagation

We develop a MAP inference algorithm for the *Probit GP-LMM*. In the following, we employ the "Heaviside likelihood view" (cf. Eq. 6.2) of the probit model and write

$$p(y_i|\beta_i, f_i) = \mathbb{1}[y_i\boldsymbol{\beta}^\top\boldsymbol{x}_i + y_i f_i \geq 0].$$

Note that without loss of generality, we can always assume this form by replacing the original kernel matrix K by $K \leftarrow K + I$ (see e.g. Bishop, 2006).

For the sake of a simpler notation and without loss of generality, we assume in this section that *all observed binary labels y_i are 1.* The reason why this assumption is not a constraint is that we can always perform a linear transformation to absorb the sign of the labels into the data matrix X and kernel matrix K. Hence, in this section we replace the data matrix by $X \leftarrow X\text{diag}(\boldsymbol{y})$ and kernel matrix by $K \leftarrow \text{diag}(\boldsymbol{y})K\text{diag}(\boldsymbol{y})$. Thus, when working with this transformed data matrix and kernel

matrix, our assumption is satisfied and we can write the likelihood as

$$p(y_i|\boldsymbol{\beta}, f_i) = \mathbb{1}[\boldsymbol{\beta}^\top \boldsymbol{x}_i + f_i \geq 0]. \tag{6.3}$$

Maximum a posteriori (MAP) inference. The goal is to compute the maximum a posteriori (MAP) estimate of the linear effect $\boldsymbol{\beta}$. We aim to find the maximizer $\hat{\boldsymbol{\beta}}$ of the objective function $\mathcal{L}(\boldsymbol{\beta}) = \log p(\boldsymbol{\beta}|\boldsymbol{y})$ and obtain the optimization problem

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \mathcal{L}(\boldsymbol{\beta}) = \arg\max_{\boldsymbol{\beta}} \log p(\boldsymbol{y}|\boldsymbol{\beta}) + \lambda_0 ||\boldsymbol{\beta}||_1. \tag{6.4}$$

The log Laplace prior translates into a ℓ_1-norm regularizer, which is also known as a Lasso regularizer. The marginal likelihood term can be expressed as

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{\beta}) &= \int \prod_{i=1}^{n} p(y_i|\boldsymbol{\beta}, f_i) \mathcal{N}(\boldsymbol{f}|0, K) \mathrm{d}\boldsymbol{f} \\ &= \int \mathbb{1}\left[X\boldsymbol{\beta} + \boldsymbol{f} \geq 0\right] \mathcal{N}(\boldsymbol{f}|0, K) \mathrm{d}\boldsymbol{f} \\ &= \int \underbrace{\mathbb{1}\left[\boldsymbol{f} \geq 0\right] \mathcal{N}(\boldsymbol{f}|X\boldsymbol{\beta}, K)}_{=p(\boldsymbol{y}, \boldsymbol{f}|\boldsymbol{\beta})} \mathrm{d}\boldsymbol{f}. \end{aligned}$$

where we used that $\boldsymbol{f}$ is symmetric in zero. This is just the multivariate Gaussian, truncated and normalized to the positive orthant. The objective (6.4) consists of optimizing the log-likelihood term in the presence of a ℓ_1-norm regularizer. Although the log-likelihood is intractable, we prove in Appendix D.1, that it is convex in $\boldsymbol{\beta}$. This allows us to apply convex optimization techniques.

Expectation maximization. Latent variable models of the type of Eq. 6.4 can be solved using expectation maximization (EM) algorithms (Dempster, Laird, and Rubin, 1977) that alternate between a gradient step in the global parameter $\boldsymbol{\beta}$ (M-step) and a Bayesian inference step (E-step) to infer the distribution over the latent variable $\boldsymbol{f}$. In our case, the E-step relies on approximate inference, which is why our approach can be called an *approximate* EM algorithm.

In more detail, to follow the gradients and optimize the (convex) objective, we employ the alternating direction method of multipliers (ADMM) in the M-step. Below, we derive analytic expressions for the Hessian and the gradient of the marginal likelihood in terms of moments of the posterior distribution over the latent noise. The inner loop (the E-step) then consists of approximating these moments by means of expectation propagation (EP).

Next, we introduce the ADMM algorithm for the M-step and then describe the EP method for the E-step.

M-step via ADMM. In the optimization problem (6.4), we encounter the problem of minimizing a convex function $\ell(\boldsymbol{\beta})$ in the presence of an additional ℓ_1-norm regularizer,

$$\mathcal{L}(\boldsymbol{\beta}) = \ell(\boldsymbol{\beta}) + \lambda_0 ||\boldsymbol{\beta}||_1. \tag{6.5}$$

The ℓ_1-norm in the objective function is not differentiable and thus prevents us from applying standard gradient-based methods such as Newton's method. This is a well-known problem and several alternative solutions have been developed; one of these is the alternating direction method of multipliers (ADMM) (Boyd et al., 2011). In ADMM, we augment the objective with the additional parameters $\boldsymbol{z}$ and $\boldsymbol{\eta}$,

$$\mathcal{L}(\boldsymbol{\beta}, \boldsymbol{z}, \eta) := -\ell(\boldsymbol{\beta}) + \lambda ||\boldsymbol{z}||_1^1 + \boldsymbol{\eta}^\top (\boldsymbol{\beta} - \boldsymbol{z}) + \tfrac{1}{2} c ||\boldsymbol{\beta} - \boldsymbol{z}||_2^2. \tag{6.6}$$

This objective can be viewed as the Lagrangian associated with the problem

$$\begin{aligned} &\min_{w,z} \; -\ell(\boldsymbol{\beta}) + \lambda ||\boldsymbol{z}||_1^1 + \tfrac{1}{2} c ||\boldsymbol{\beta} - \boldsymbol{z}||_2^2 \\ &\text{s.t. } \boldsymbol{z} = \boldsymbol{\beta}, \end{aligned}$$

which is equivalent to the original problem (6.5). Since strong duality holds we can solve the primal problem in Eq. 6.5 by solving the dual problem, Eq. 6.6. This is done by an iterative scheme where we alternate between the minimization updates for $\boldsymbol{\beta}$ and z and a gradient step in η. Note that the term $\frac{1}{2} c ||\boldsymbol{\beta} - \boldsymbol{z}||_2^2$ is optional but grants better numerical stability and faster convergence. Details on the ADMM algorithm can be found in Boyd et al., 2011. Note that also other optimization methods are possible, which deal with non-smooth objectives such as ours, in particular, subgradient methods. The benefit of the ADMM approach, though, is that it allows us to use second-order information because the objective is now smooth in $\boldsymbol{\beta}$.

Approximate E-step via expectation propagation. The inner loop of the EM algorithm amounts to computing the gradient and Hessian of $\mathcal{L}(\boldsymbol{\beta}, z, \eta)$ w.r.t. $\boldsymbol{\beta}$. Computing derivatives of the ℓ_2-norm regularizer is trivial and we, therefore, focus on the log-likelihood $\ell(\boldsymbol{\beta})$. The gradient and Hessian of $\ell(\boldsymbol{\beta})$ are not available in closed form. However, we will show that they can be expressed in terms of the first and second moment of the complete conditional distribution $p(\boldsymbol{f}|\boldsymbol{y}, \boldsymbol{\beta})$.

In the following, we use the shorthand notation

$$\boldsymbol{\mu} \;\equiv\; \boldsymbol{\mu}(\boldsymbol{\beta}) \;=\; X\boldsymbol{\beta}.$$

The complete conditional distribution of the latent GP is

$$p(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{\beta}) = \frac{p(\boldsymbol{y},\boldsymbol{f}|\boldsymbol{\beta})}{\int p(\boldsymbol{y},\boldsymbol{f}|\boldsymbol{\beta})\mathrm{d}\boldsymbol{f}} = \frac{\mathbb{1}\left[\boldsymbol{f}\geq 0\right]\mathcal{N}(\boldsymbol{f}|\boldsymbol{\mu},K)}{\int_{\mathbb{R}^n_+}\mathcal{N}(\boldsymbol{f};\boldsymbol{\mu},K)\,\mathrm{d}\boldsymbol{f}}.$$

We furthermore introduce

$$\begin{aligned}\boldsymbol{\mu}_p(\boldsymbol{\beta}) &= \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{\beta})}\left[\boldsymbol{f}\right], \\ \Sigma_p(\boldsymbol{\beta}) &= \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{\beta})}\left[(\boldsymbol{f}-\boldsymbol{\mu}_p(\boldsymbol{\beta}))(\boldsymbol{f}-\boldsymbol{\mu}_p(\boldsymbol{\beta}))^\top\right].\end{aligned} \tag{6.7}$$

This is just the mean and the covariance of the *truncated* multivariate Gaussian $p(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{\beta})$, as opposed to $\boldsymbol{\mu}, K$, which are the mean and covariance of the non-truncated Gaussian $\mathcal{N}(f|\boldsymbol{\mu},K)$. In general, these expectations do not have a closed-form solution. However, we develop suitable approximations for them in the following.

We abbreviate $\boldsymbol{\mu}_p \equiv \boldsymbol{\mu}_p(\boldsymbol{\beta})$ and $\Sigma_p \equiv \Sigma_p(\boldsymbol{\beta})$ and write $\Delta\boldsymbol{\mu} = \boldsymbol{\mu}_p - \boldsymbol{\mu}$ for the difference between the means of the posterior (the truncated Gaussian) and the un-truncated Gaussian. The gradient and Hessian of $\ell(\boldsymbol{\beta})$ are given by

$$\begin{aligned}\nabla_{\boldsymbol{\beta}}\ell(\boldsymbol{\beta}) &= -\Delta\boldsymbol{\mu}K^{-1}X, \\ \nabla^2_{\boldsymbol{\beta}}\ell(\boldsymbol{\beta}) &= X^\top[K^{-1}(\Sigma_p - \Delta\boldsymbol{\mu}\Delta\boldsymbol{\mu}^\top)K^{-1} - K^{-1}]X.\end{aligned} \tag{6.8}$$

Proofs are given in Appendix D.2. Note that the variable $\boldsymbol{\beta}$ enters through $\Sigma_p(\boldsymbol{\beta})$ and $\Delta\boldsymbol{\mu}(\boldsymbol{\beta})$.

The next step is to approximate the quantities $\boldsymbol{\mu}_p$ and Σ_p in Eq. 6.7, which we need for computing Eq. 6.8. These are intractable, involving expectations over the posterior of $\boldsymbol{f}$. Hence, we use approximate Bayesian inference methods to obtain estimates of these expectations.

We employ expectation propagation (EP) (Minka, 2001; Opper and Winther, 2001) to approximate the moments of truncated Gaussian integrals using the approach proposed by Cunningham, Hennig, and Lacoste-Julien (2011).

EP approximates the posterior $p(\boldsymbol{f}|\boldsymbol{y},\boldsymbol{\beta})$ in terms of a variational distribution $q(\boldsymbol{f})$. The EP objective function is involved, but EP is motivated by the idea of minimizing the (reverse) Kullback-Leibler divergence from the true posterior to the approximate distribution (Cunningham, Hennig, and Lacoste-Julien, 2011). In the Gaussian EP method proposed by Cunningham, Hennig, and Lacoste-Julien, 2011, the variational distribution $q^*(\boldsymbol{f})$ is an un-truncated Gaussian, characterized by the variational parameters $\boldsymbol{\mu}_{q^*}$ and Σ_{q^*}. We approximate the posterior in terms of the variational distribution, whose mean and covariance are $\boldsymbol{\mu}_p \approx \boldsymbol{\mu}_{q^*}$ and $\Sigma_p \approx \Sigma_{q^*}$. We

warm-start each gradient computation with the optimal parameters of the earlier iteration. As a remark, instead of computing the first and second moment of the integral to compute the gradient and Hessian, $\ell(\beta)$ could also be optimized numerically e.g. using BFGS (Fletcher, 1987) where the integral is still approximated using EP. This is less efficient as it requires many evaluations of the integral for a single gradient estimate.

Alg. 3 summarizes our procedure. We denote the expectation propagation algorithm for approximating the first and second moment of the truncated Gaussian by $\mathrm{EP}(\boldsymbol{\mu}, K)$. Here, $\boldsymbol{\mu}$ and K are the mean and covariance matrix of the un-truncated Gaussian. The subroutine returns the first and second moments of the truncated distributions $\boldsymbol{\mu}_q$ and Σ_q. When initialized with the outcomes of earlier iterations, this subroutine typically converges within a single EP loop.

Algorithm 3 Approximate Inference for the Probit-LMM

repeat
  initialize $\beta = \beta^k$
  repeat
    $(\mu_q, \Sigma_q) \leftarrow \mathrm{EP}(\beta)$
    $\Delta\mu = \mu_q - X\beta$
    $g = \Delta\mu^\top K^{-1} X + c(\beta - z^k + \eta^k)^\top$
    $H = X^\top [K^{-1} - K^{-1}(\Sigma_q - \Delta\mu\Delta\mu^\top)K^{-1}]X + cI$
    $\beta = \beta - \alpha_t H^{-1} g$
  until criterion 2 is met
  `# ADMM updates`
  $\beta^{k+1} = \beta$
  $z^{k+1} = S_{\lambda/c}(\beta^{k+1} + \eta^k)$ `# soft thresholding, see Boyd et al. (2011)`
  $\eta^{k+1} = \eta^k + \beta^{k+1} - z^{k+1}$
until criterion 1 is met

Our algorithm thus consists of two nested loops; the outer ADMM loop, containing the Newton update, and the inner EP loop, which computes the moments of the posterior. We choose stopping *criterion 1* to be the convergence criterion proposed by Boyd (Boyd et al., 2011) and choose *criterion 2* to be always fulfilled, i. e. we perform only one Newton optimization step in the inner loop. Our experiments showed that doing only one Newton optimization step, instead of executing until convergence, is stable and leads to significant improvements in speed. ADMM is known to converge even when the minimizations in the ADMM scheme are not carried out exactly (see e.g. Eckstein and Bertsekas, 1992).

Predictions. To predict the label y_* of a test point $\boldsymbol{x}_*$ we substitute the approximate posterior distributions of the latent GP $q(\boldsymbol{f})$ and the point estimate $\hat{\boldsymbol{\beta}}$ into the standard predictive distribution and obtain

$$\begin{aligned} p(y_*|\boldsymbol{x}_*) &\approx \int p(y_*|\hat{\boldsymbol{\beta}}, f_*)p(f_*|\boldsymbol{f})q(\boldsymbol{f})\mathrm{d}f_*\mathrm{d}\boldsymbol{f} \\ &= \int p(y_*|\hat{\boldsymbol{\beta}}, f_*)\mathcal{N}\left(f^*|\mu_*,\ \sigma_*^2\right)\mathrm{d}f_*, \end{aligned} \tag{6.9}$$

with

$$\begin{aligned} \mu_* &= \boldsymbol{K}_{*n}K_{nn}^{-1}\boldsymbol{\mu}_q \\ \sigma_*^2 &= K_{**} + \boldsymbol{K}_{*n}K_{nn}^{-1}\left(\Sigma_q K_{nn}^{-1} - I\right)\boldsymbol{K}_{n*} \end{aligned}$$

The matrix $\boldsymbol{K}_{*n}$ denotes the kernel matrix between the test point and the training points and K_{**} the kernel value of the test point. This one-dimensional integral can be easily computed by numerical quadrature methods.

6.5 Inference via augmented variational inference

In this section, we develop a fully Bayesian inference method for the logistic GP linear mixed model building on variational inference. We first augment the intractable logistic likelihood and Laplace prior to obtain a conditionally conjugate model. As outlined in Chapter 2, this allows us to compute efficient variational inference updates in closed form. Finally, we describe how this approach can also be used in a pure MAP estimation setting. In this section, we work with the original formulation of the model given in Eq. 6.1, i.e. we consider the original training labels $y_i \in \{-1, 1\}$ and do not absorb the labels into the data matrix X and kernel matrix K.

The augmentation. In the following, we propose an augmentation that leads to a conditionally conjugate model. We first augment the likelihood, such that it becomes Gaussian in the latent GP values $\boldsymbol{f}$ and linear effect $\boldsymbol{\beta}$. We then augment the Laplace prior on $\boldsymbol{\beta}$ such it becomes Gaussian too.

First, we employ a Pólya-Gamma augmentation approach to the logistic likelihood similar to the augmentation for the GP classification model in Chapter 3 and obtain

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{f}, \boldsymbol{\omega}) &= \prod_i p(y_i|\boldsymbol{\beta}, f_i, \omega_i) \\ &\propto \exp\left\{-\frac{1}{2}(-y^\top(X\boldsymbol{\beta} + \boldsymbol{f}) + (X\boldsymbol{\beta} + \boldsymbol{f})^\top\Omega(X\boldsymbol{\beta} + \boldsymbol{f}))\right\}, \end{aligned} \tag{6.10}$$

where $\Omega = \text{diag}(\boldsymbol{\omega})$ and $p(\omega_i) = \text{PG}(\omega_i|0,1)$

The likelihood is now Gaussian in $\boldsymbol{f}$ and $\boldsymbol{\beta}$ and, hence, conjugate in $\boldsymbol{f}$. However, the Laplace prior of $\boldsymbol{\beta}$ still imposes a non-conjugate structure. We circumvent the problem by introducing an additional augmentation for the Laplace prior building on the integral identity

$$\begin{aligned} p(\boldsymbol{\beta}) &= \frac{1}{2}\lambda_0 \exp(\lambda_0|\boldsymbol{\beta}|) = \frac{1}{2}\prod_{j=1}^{d} \lambda_0 \exp(\lambda_0|\beta_j|) \\ &= \prod_j \int p(\beta_j|\lambda_j)p(\lambda_j)\mathrm{d}\lambda_j, \end{aligned}$$

with distributions

$$\begin{aligned} p(\beta_j|\lambda_j) &= \mathcal{N}(\beta_j|0, \frac{1}{\lambda_0\lambda_j}) \\ p(\lambda_j) &= \frac{4}{\lambda_j^2}\exp(-\frac{1}{2\lambda_j}). \end{aligned} \tag{6.11}$$

Augmenting our model with variables $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_d)$ as defined above in Eq. 6.11, leads to a Gaussian prior over $\boldsymbol{\beta}$. The final augmented model is fully conditionally conjugate and given by

$$p(\boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{f}, \boldsymbol{\omega}, \boldsymbol{\lambda}) = \prod_i p(y_i|\boldsymbol{\beta}, f_i, \omega_i)p(\omega_i)p(f) \prod_j p(\beta_j|\lambda_j)p(\lambda_j).$$

Complete conditional distributions. The complete conditional distribution of the latent GP $\boldsymbol{f}$ is

$$\begin{aligned} p(\boldsymbol{f}|\boldsymbol{\beta}, \boldsymbol{\omega}, \boldsymbol{y}) &\propto p(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{f}, \boldsymbol{\omega})p(f) \\ &\propto \exp\left\{-\frac{1}{2}(-\boldsymbol{y}^\top(X\boldsymbol{\beta}+\boldsymbol{f}) + (X\boldsymbol{\beta}+\boldsymbol{f})^\top\Omega(X\boldsymbol{\beta}+\boldsymbol{f}))\right\}\mathcal{N}(\boldsymbol{f}|0,K) \\ &\propto \mathcal{N}(\boldsymbol{f}|\boldsymbol{\mu}_f, \Sigma_f), \end{aligned}$$

with

$$\boldsymbol{\mu}_f = \Sigma_f\left(\frac{1}{2}\boldsymbol{y} - \Omega X\boldsymbol{\beta}\right), \quad \Sigma_f = (\Omega + K^{-1})^{-1}.$$

and K is the kernel matrix evaluated at the training points.

The complete conditional distribution of $\boldsymbol{\beta}$ is

$$\begin{aligned} p(\boldsymbol{\beta}|\boldsymbol{f}, \boldsymbol{\omega}, \boldsymbol{y}) &\propto \exp\left\{-\frac{1}{2}(-\boldsymbol{y}^\top(X\boldsymbol{\beta}+\boldsymbol{f}) + (X\boldsymbol{\beta}+\boldsymbol{f})^\top\Omega(X\boldsymbol{\beta}+\boldsymbol{f}))\right\} \\ &\quad \prod_j \mathcal{N}\left(\beta_j|0, \frac{1}{\lambda_0\lambda_j}\right) \\ &\propto \mathcal{N}\left(\boldsymbol{\beta}|\boldsymbol{\mu}_\beta, \Sigma_\beta\right), \end{aligned}$$

with

$$\boldsymbol{\mu}_\beta = \Sigma_\beta \left(\frac{1}{2} X^\top \boldsymbol{y} - X\Omega \boldsymbol{f} \right), \quad \Sigma_\beta = (X^\top \Omega X + \lambda_0 \Lambda)^{-1},$$

where $\Lambda := \mathrm{diag}(\boldsymbol{\lambda})$.

The complete conditional distribution of λ_j is

$$\begin{aligned} p(\lambda_j | \beta_j) &\propto \mathcal{N}\left(\beta_j | 0, \frac{1}{\lambda_0 \lambda_j}\right) \frac{4}{\lambda_j^2} \exp\left(-\frac{1}{2\lambda_j}\right) \\ &\propto \lambda_j^{-\frac{3}{2}} \exp\left(-\frac{1}{2}(\beta_j^2 \lambda_j \lambda_0 + \lambda_j^{-1})\right) \\ &\propto \mathrm{GIG}\left(\lambda_j \mid \lambda_0 \beta_j^2, 1, -\frac{1}{2}\right). \end{aligned}$$

Note that $p(\lambda_j|\beta_j)$ could also be viewed as an inverse Gaussian distribution, which is a subset of the generalized inverse Gaussian family. But in the following, we still use the generalized inverse Gaussian notation.

The complete conditional distribution of ω_i is

$$\begin{aligned} p(\omega_i | \boldsymbol{\beta}, f_i, y_i) &\propto \exp\left(-\frac{1}{2}(\boldsymbol{\beta}^\top \boldsymbol{x}_i + f_i)^2 \omega_i\right) \mathrm{PG}(\omega_i | 1, 0) \\ &\propto \mathrm{PG}(\omega_i \mid 1,\ \boldsymbol{\beta}^\top \boldsymbol{x}_i + f_i), \end{aligned}$$

where we used the exponential tilting property of Pólya-Gamma variables.

Variational approximation. To scale our model to big datasets, we approximate the latent GP f by a *sparse GP* approximation building on *inducing points*. We introduce M inducing points $\boldsymbol{u}$ and connect the GP values with the inducing points via the joint prior distribution $p(\boldsymbol{f}, \boldsymbol{u})$ as outlined in Section 1.2.

We follow a structured mean field approach (Wainwright and Jordan, 2008) and assume a decoupled variational distribution $q(\boldsymbol{\beta}, \boldsymbol{u}, \boldsymbol{\omega}, \boldsymbol{\lambda}) = q(\boldsymbol{\beta}) q(\boldsymbol{u}) q(\boldsymbol{\omega}) q(\boldsymbol{\lambda})$. Since our model is conditionally conjugate, the family of the optimal variational distribution can be easily determined by averaging the complete conditionals in log-space (see e.g. Blei, Kucukelbir, and McAuliffe, 2017 and Section 1.2). We obtain the variational family

$$\begin{aligned} q(\boldsymbol{\beta}) &= \mathcal{N}(\boldsymbol{\beta} | \boldsymbol{\mu}_\beta, \Sigma_\beta) & q(\boldsymbol{u}) &= \mathcal{N}(\boldsymbol{u} | \boldsymbol{\mu}_u, \Sigma_u), \\ q(\boldsymbol{\lambda}) &= \prod_j \mathrm{GIG}\left(\lambda_j \mid a_j, 1, -1/2\right) & q(\boldsymbol{\omega}) &= \prod_i \mathrm{PG}(\omega_i \mid 1,\ c_i). \end{aligned}$$

The mean field optimality conditions reveal that the variational distributions $q(\lambda_j)$, $q(\omega_i)$ have only one free variational parameter a_j and c_i, respectively.

Inference method. Building on the conditional conjugate augmentation, we develop an *augmented variational inference* algorithm as outlined in Section 2.2.

By applying the standard coordinate ascent update formula given in Hoffman et al., 2013 to each variational distributions, we obtain the closed-form update for each variational parameter,

$$\begin{aligned}
\boldsymbol{\mu}_u =& \Sigma_u \left(\frac{1}{2} \kappa^\top \boldsymbol{y} - \kappa^\top \bar{\Omega} X \boldsymbol{\mu}_\beta \right) \\
\Sigma_u =& \left(\kappa^\top \bar{\Omega} \kappa + K_{mm}^{-1} \right)^{-1} \\
\boldsymbol{\mu}_\beta =& \Sigma_\beta \left(\frac{1}{2} X^\top \boldsymbol{y} - X \bar{\Omega} \kappa \boldsymbol{\mu}_u \right) \\
\Sigma_\beta =& (X^\top \bar{\Omega} X + \lambda_0 \bar{\Lambda})^{-1}, \\
c_i =& \sqrt{\boldsymbol{x}_i^\top \Sigma_\beta \boldsymbol{x}_i + \left(\boldsymbol{x}_i^\top \boldsymbol{\mu}_\beta + \kappa \mu_{u_i} \right)^2 + \kappa \Sigma_u \kappa^\top + \tilde{K}_{ii}} \\
a_j =& \lambda_0 \left(\mu_{\beta_j}^2 + (\Sigma_\beta)_{jj} \right),
\end{aligned}$$

where $\kappa = K_{nm} K_{mm}^{-1}$ and $\widetilde{K} = K_{nn} - K_{nm} K_{mm}^{-1} K_{mn}$. The expectations $\bar{\Lambda}$ and $\bar{\Omega}$ are computed in closed form,

$$\begin{aligned}
\bar{\Lambda} &= \mathbb{E}_{q(\boldsymbol{\lambda})}[\Lambda] = \frac{1}{\sqrt{a_j}} \\
\bar{\Omega} &= \mathbb{E}_{q(\boldsymbol{\omega})}[\Omega] = \operatorname{diag}\left(\frac{1}{2c_i} \tanh\left(\frac{c_i}{2} \right) \right).
\end{aligned}$$

When using mini-batches of the data, each global variational parameter (i.e. $\boldsymbol{\mu}_u$, Σ_u, $\boldsymbol{\mu}_\beta$ and Σ_β) is updated using a convex combination of the old parameter and the CAVI update, which corresponds to a natural gradient ascent scheme (see Section 2.2).

Predictions. To predict the label y_* of a test point $\boldsymbol{x}_*$ we substitute the approximate posterior distributions into the standard predictive distribution and obtain

$$\begin{aligned}
p(y_* | \boldsymbol{x}_*) &\approx \int p(y_* | \boldsymbol{\beta}, f_*) p(f_* | \boldsymbol{u}) q(\boldsymbol{u}) q(\boldsymbol{\beta}) \mathrm{d}f_* \mathrm{d}\boldsymbol{u} \mathrm{d}\boldsymbol{\beta} \\
&= \int p(y_* | \boldsymbol{\beta}, f_*) \mathcal{N}\left(f^* | \mu_*, \sigma_*^2 \right) \mathrm{d}f_*, \qquad (6.12)
\end{aligned}$$

with

$$\begin{aligned}
\mu_* &= \boldsymbol{x}_*^T \boldsymbol{\mu}_\beta + \boldsymbol{K}_{*m} K_{mm}^{-1} \boldsymbol{\mu}_u \\
\sigma_*^2 &= K_{**} + \boldsymbol{K}_{*m} K_{mm}^{-1} \left(\Sigma_u K_{mm}^{-1} - I \right) \boldsymbol{K}_{m*} + \boldsymbol{x}_*^T \Sigma_\beta \boldsymbol{x}_*.
\end{aligned}$$

The matrix $\boldsymbol{K}_{*m}$ denotes the kernel matrix between the test point and the inducing points and K_{**} the kernel value of the test point. This one-dimensional integral can be easily computed by numerical quadrature methods.

MAP inference via variational inference. If we are only interested in the MAP estimate of $\boldsymbol{\beta}$, we could just use the mean of the optimal variational distribution $q(\boldsymbol{\beta})$ obtained by the fully Bayesian inference method described above. As an alternative, we can bypass the augmentation of the Laplace prior and apply an approximate expectation maximization approach similar to the EP method from the previous section. We build on the ADMM method, which optimizes the augmented Lagrangian objective (6.6) with a slight modification. The only difference is that we now optimize the intractable log-likelihood term $\ell(\boldsymbol{\beta})$ using a variational expectation maximization approach.

To this end, we use the Pólya-Gamma augmented model (see Eq. 6.10), but without the augmentation of Laplace prior. We employ variational approximations of the posterior of the GP and Pólya-Gamma variables $q(\boldsymbol{f})$ and $q(\boldsymbol{\omega})$ and obtain the variational lower bound

$$\begin{aligned}\ell(\boldsymbol{\beta}) &= \log p(\boldsymbol{y}|\boldsymbol{\beta}) \\ &\geq \mathbb{E}_{q(\boldsymbol{f})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}, \boldsymbol{f}|\boldsymbol{\beta}) - \log q(\boldsymbol{f}) - \log q(\boldsymbol{\omega})],\end{aligned}$$

which can be computed and optimized in closed form (using updates which are similar to the fully Bayesian case). We alternate between optimization steps in $\boldsymbol{\beta}$ and updating the variational parameters of $q(\boldsymbol{f})$ and $q(\boldsymbol{\omega})$.

6.6 Experiments

We study the performance of our proposed methods in experiments on both artificial and real-world data. We consider the two versions of our model—PROBIT GP-LMM, which is described in Section 6.4 and trained via expectation propagation and LOGISTIC GP-LMM, which is described in Section 6.5 and trained via *augmented variational inference*. Our data are taken from the domains of statistical genetics and computer malware prediction.

We compare our algorithms against three competing methods, including sparse PROBIT REGRESSION and GP CLASSIFICATION, which are limiting cases of our model and are trained with the corresponding parts of the algorithm discussed in Section 6.4.

As a third related method, we compare against LMM-LASSO, which is trained via the algorithm proposed by Lippert et al., 2011.

In all considered cases, our proposed GP-LMM models achieve higher classification performance whereby the PROBIT GP-LMM has slightly better performance on small datasets. Also, the features that our algorithms find are less affected by spurious correlations induced by population structure. Furthermore, we demonstrate that the LOGISTIC GP-LMM scales to big datasets containing more than 100,000 instances.

6.6.1 General experimental setup

For the real-world and synthetic experiments, we first need to make a choice for the class of kernels that we use for the covariance matrix. We choose a combination of three contributions,

$$K = \lambda_1 I + \lambda_2 X X^\top + \lambda_3 K_{\text{side}}. \tag{6.13}$$

We use a contribution of an identity matrix and a linear kernel, which is known to work well on genetic data (Lippert et al., 2011; Lippert, 2013). The third term is optional and depends on the context; it is a kernel representing any side information provided in an auxiliary feature matrix X'. Here, we compute K_{side} as a squared exponential kernel from the side information X'. Unless stated otherwise, we evaluate the methods in all experiments by using n individuals from the dataset for training and splitting the remaining dataset equally into validation and test sets. This process is repeated 50 times, over which we report on average accuracies or areas under the ROC curve (AUCs), as well as standard errors (Fawcett, 2006).

The hyperparameters of the kernels (i.e. λ_1, λ_2, λ_3 and the length scale parameter l of the squared exponential kernel), together with the regularization parameter λ_0, were determined on the validation set, using grid search over a sufficiently large parameter space (optimal values are attained inside the grid; in most cases $\lambda_i, l \in [0.1, 1000]$). For all datasets, the features were centered and scaled to unit standard deviation, except in experiment 6.6.4, where the features are binary.

In Sections 6.6.3 and 6.6.4, we show that including a linear kernel into the covariance matrix leads to top-ranked features, which are less correlated with the population structure in comparison to the top-ranked features of sparse PROBIT REGRESSION. The correlation plots[3] in Fig. 6.5 show the mean correlation of the top features with

[3]The correlation plots in Fig. 6.5 are created according to Li, Rakitsch, and Borgwardt, 2011 as follows. First, we randomly choose 70% of the available data as training set and obtain a weight vector w by training. We compute the empirical Pearson correlation coefficient of each feature with

population structure and the corresponding standard errors. All experiments were performed on a Linux machine with 48 CPU kernels (each 2.4GHz) and 368GB RAM.

6.6.2 Simulated data

To test the properties of our model in a controlled setup, we first generate synthetic data as follows. We generate a weight vector $\boldsymbol{\beta} \in \mathbb{R}^d$ with $1 \leq k \leq d$ entries being 1, and the other $d - k$ entries being 0. We choose $d = 50$ and vary k. We then create a random covariance matrix $K_{\text{side}} \in \mathbb{R}^{n \times n}$, which serves as side information matrix[4]. We sample $n = 200$ points $X = \{\boldsymbol{x}_1, \dots \boldsymbol{x}_n\}$ independently from a uniform distribution over the unit cube $[-1, 1]^d$ and create the labels according to the PROBIT GP-LMM model, Eq. 6.1, using K_{side} as covariance matrix. We reserve 100 samples for training and 50 for validation and testing, respectively.

The synthetic data allows us to control the sparsity level k of non-zero features. We then fit various models to the data to predict the binary labels: PROBIT GP-LMM (proposed) as well as LOGISTIC GP-LMM (proposed), GP CLASSIFICATION, the LMM-LASSO and standard ℓ_1-norm regularized (sparse) PROBIT REGRESSION. As a benchmark, we introduce the ORACLE classifier, where we use the ground truth model for prediction.

Fig. 6.1 shows the resulting accuracies. The horizontal axis shows the varying percentage of non-zero features in the artificial data k/d. Note that the accuracies of all methods fluctuate due to the finite size of the different datasets that we generated.

The observed performances of the methods depend on the varying level of the sparsity of the data: if the true linear effect is sparse, sparsely regularized models should be expected to work better. The opposite can be expected from models that include all features in a dense way, such as GP CLASSIFICATION. These models are good when the true effects are dense. Our plot indeed reveals this tendency. ℓ_1-norm regularized (sparse) PROBIT REGRESSION performs well for small k, whereas GP CLASSIFICATION works well for large k. Both versions of our GP-LMM model

the first principle component of the linear kernel on top of the data. This is a way to measure the correlation with the population structure (Price et al., 2006). We define the index set I by taking the absolute value of each entry of w and sorting them in descending order. We now sort the so-obtained list of correlation coefficients with respect to the index set I and obtain a resorted list of correlation coefficients $(c_1, \dots, c_n)$. In the last step, we obtain a new list $(\hat{c}_1, \dots, \hat{c}_n)$ by smoothing the values, computing $\hat{c}_i := \frac{1}{i} \sum_k^i c_k$. Finally, we plot the values $(\hat{c}_1, \dots, \hat{c}_n)$ with respect to I. This procedure was repeated 30 times for different random choices of training sets.

[4] The covariance matrix was created as follows. The random generator in MATLAB version 8.3.0.532 was initialized to seed = 20 using the `rng(20)` command. The matrix K_{side} was realized in two steps via `A=2*rand(50,200)-1` and K_{side}`=3*A'*A+0.6*eye(200)+3*ones(200,200)`.

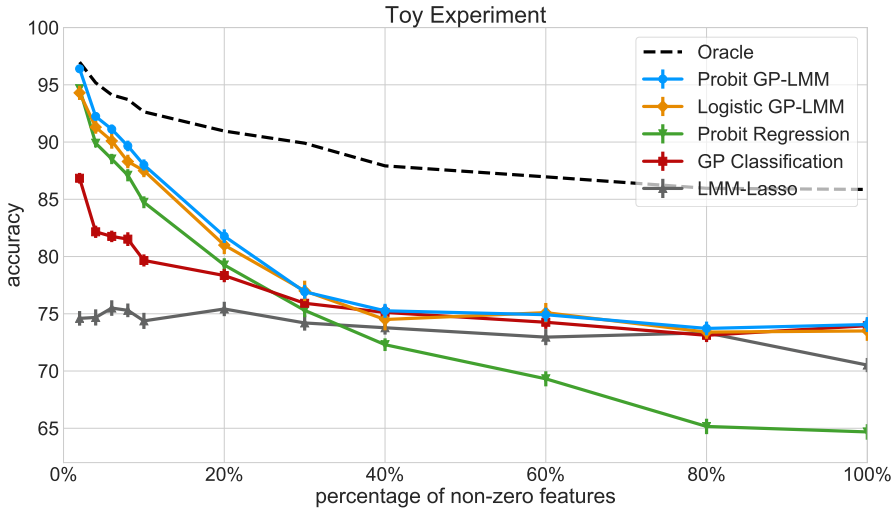

Figure 6.1: TOY: Average accuracies (in %) as a function of the percentage of true non-zero features in the generating model. The (small) error bars indicate one standard deviation. (Proposed methods: PROBIT GP-LMM and LOGISTIC GP-LMM.)

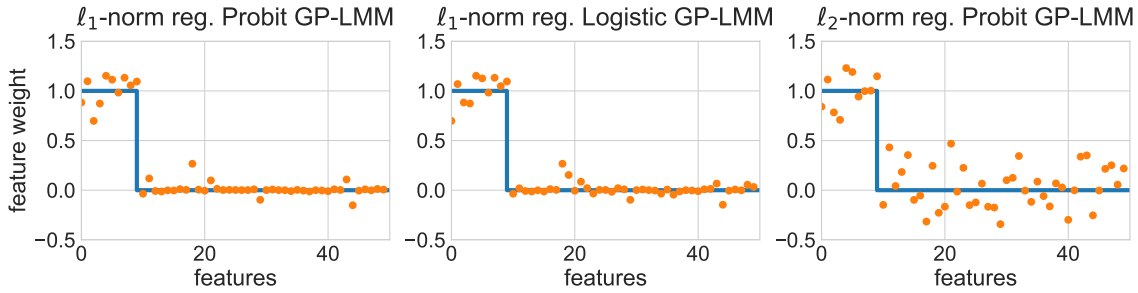

Figure 6.2: TOY: Effects of the regularizer on the model's ability to select features. Ground truth (blue solid line) and feature weights (orange dots) of ℓ_1-norm regularized PROBIT GP-LMM and LOGISTIC GP-LMM vs. ℓ_2-norm regularized PROBIT GP-LMM. The ℓ_1-norm regularized versions of our model attain better sparsity by setting the weights of irrelevant features closer to zero.

outperform the competing methods, because they contain both a dense kernel as well as a sparse linear effect. The PROBIT GP-LMM performs naturally slightly better than the LOGISTIC GP-LMM since the labels in the synthetic data are created using the probit likelihood (and not the logistic likelihood). Interestingly, even though the LMM-LASSO also has a sparse effect and a dense kernel, its performance is not very compelling on our experimental dataset. This may be explained by its output being continuous (and not binary) and, therefore, not well suited for classification tasks.

Finally, we analyze the importance of the ℓ_1-norm regularizer in the GP-LMM model and compared it against a model that is ℓ_2-regularized. In particular, we compare the ℓ_1-norm regularized PROBIT GP-LMM and LOGISTIC GP-LMM against the ℓ_2-norm regularized version of the PROBIT GP-LMM. We generate an artificial dataset with $k = 10$ non-zero features and try to recover these non-zero feature weights with all three methods. Fig. 6.2 shows the results of this analysis. The blue solid line represents the truly non-zero weights, while the orange dots show our estimates when using ℓ_1-norm (left and middle) and ℓ_2-norm (right) regularization on $\boldsymbol{\beta}$, respectively. We observe that the ℓ_1-norm regularized models find better estimates of the linear weight vectors that were used to generate the data.

6.6.3 Tuberculosis disease outcome prediction

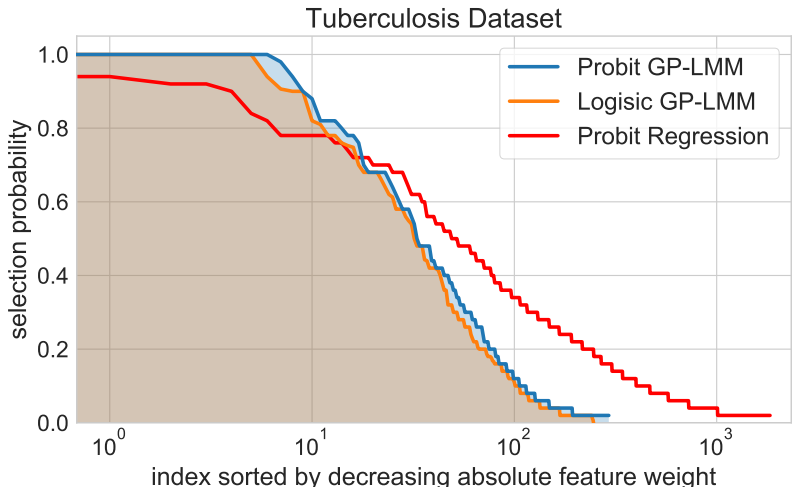

Figure 6.3: TBC: Stability of selected features for the PROBIT GP-LMM, LOGISTIC GP-LMM and sparse PROBIT REGRESSION. The plot shows the selection probabilities for each feature. Ideally, we want these to be 0 or 1. The proposed GP-LMM models lead to more stable top features and have less variability under bootstrapping.

In our first real-world experiment, we predict the outcome of tuberculosis from gene expression levels. We obtain the dataset by Berry et al., 2010 from the National Center for Biotechnology Information website[5], which includes 40 blood samples from patients with active tuberculosis as well as 103 healthy controls, together with the transcriptional signature of blood samples measured in a microarray experiment with 48,803 gene expression levels, which serve as features for our purposes. Also available

[5] `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE19491`

	AUC	Time
Probit GP-LMM	$\mathbf{86.4\% \pm 0.3}$	3.14 sec
Logistic GP-LMM	$86.1\% \pm 0.4$	3.57 sec
Probit Regression	$78.4\% \pm 0.2$	1.33 sec
GP Classification	$85.2\% \pm 0.2$	2.21 sec
LMM-Lasso	$81.2\% \pm 0.2$	**1.25** sec

Table 6.1: TBC: Average AUC and corresponding standard deviations attained in the tuberculosis disease outcome prediction experiment along with average training time in seconds. Both versions of our GP-LMM model perform similarly well and beat the competitors in terms of prediction performance.

is the age of the subjects when the blood sample was taken, from which we compute K_{side}[6]. We use $n = 80$ data points for training. To be consistent with previous studies (e.g. Li, Rakitsch, and Borgwardt, 2011), we report on the area under the ROC curve (AUC), rather than accuracy. The results are shown in Table 6.1.

We observe that the GP-LMM method in both versions yield a consistent improvement over sparse PROBIT REGRESSION (around 10 percentage points), GP CLASSIFICATION (by 0.9 to 1.2 percentage points) and LMM-LASSO (by around 5 percentage points). In Fig. 6.5, left, we show the correlation between the top features and the population structure (as confounding factor) for the PROBIT GP-LMM, LOGISTIC GP-LMM and sparse PROBIT REGRESSION. The plot was created as explained in Section 6.6.1. We find that the features obtained by both versions of the GP-LMM show less correlation with population structure than the features of sparse PROBIT REGRESSION. This is because population structure was built into our model as a source of correlated noise.

To make sure that our selected features are reliable, we investigate their stability under bootstrapping. We considered stability selection (Meinshausen and Bühlmann, 2010), where we randomly subsample 90% of the data 100 times (to accommodate the limited sample size, we follow Rakitsch et al., 2013 and do not use 50% of the samples for each draw as proposed in the original article). We define a feature to be selected if the absolute weight exceeds the threshold of 0.001. In Fig. 6.3, we show the selection probability for each feature. For the PROBIT GP-LMM, the top 7 features and for LOGISTIC GP-LMM, the top 6 features are selected in every singe run out of 100 runs, indicating that they are very stable. In contrast, in sparse PROBIT REGRESSION, these features only get selected with about 90% probability. Also, the

[6]We compute K_{side} as a squared exponential kernel on top of the side information age using length scale $l = 0.2$.

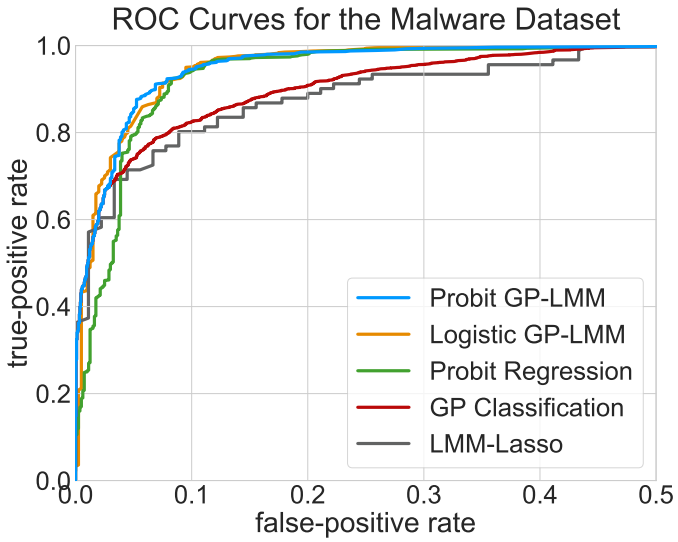

Figure 6.4: MALWARE: Average ROC curves for the computer malware detection experiment.

total number of selected features over all runs is 294 and 289 for the PROBIT GP-LMM and LOGISTIC GP-LMM, respectively; whereas for sparse PROBIT REGRESSION it is 1837. This indicates that in our models, there is less variability compared to the standard Lasso (PROBIT REGRESSION) approach. The GP-LMM models thus lead to more stable features than the standard Lasso approach.

	$AUC_{0.1}$	Time
Probit GP-LMM	$\mathbf{74.9\% \pm 0.2}$	14.89 sec
Logistic GP-LMM	$74.1\% \pm 0.3$	10.43 sec
Probit Regression	$67.2\% \pm 0.3$	8.91 sec
GP Classification	$69.8\% \pm 0.3$	8.57 sec
LMM-Lasso	$66.45\% \pm 0.3$	**5.38** sec

Table 6.2: MALWARE (SUBSET): Average $AUC_{0.1}$ (area under the ROC curve on the interval $[0, 0.1]$ and renormalized) and corresponding standard deviations attained on the 200 sample subset of the malware dataset along with average training time in seconds. Both versions of our GP-LMM model beat the competitors in terms of prediction performance whereby the PROBIT GP-LMM attains a slightly higher $AUC_{0.1}$.

6.6.4 Malicious computer software (malware) detection

We experiment on the Drebin dataset[7] (Arp et al., 2014), which contains 5,560 Android software applications from 179 different malware families. There are 545,333 binary features; each feature denotes the presence or absence of a certain source code string (such as a permission, an API call or a network address). It makes sense to look for sparse representations (Arp et al., 2014), as only a small number of strings are truly characteristic of malware. The idea is that we consider populations of different families of malware when training and hence correct for the analog of genetic population structure in this new context, that we call "malware structure".

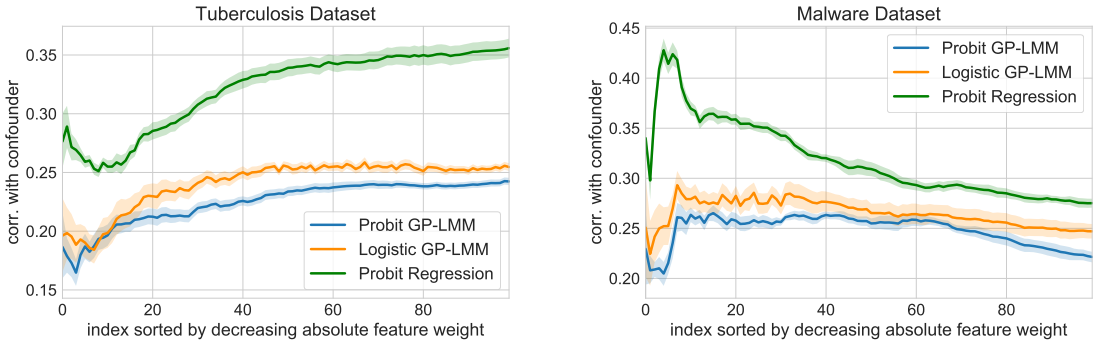

Figure 6.5: Correlation between the selected features and population structure as described in the main text (low values are better). The tuberculosis experiment is shown left and computer malware shown right. The x-axis is sorted by descending absolute weights. Shaded areas indicate standard errors. Both versions of our proposed GP-LMM model select top features that are less correlated with the confounder opposed to the features selected by PROBIT REGRESSION.

In this experiment, we concentrate on the top 10 most frequently occurring malware families in the dataset[8]. We take 10 instances from each family, forming together a malicious set of 100 and a benign set of another 100 instances (i.e., in total 200 samples). We employ $n = 80$ instances for training and stratify in the sense that we make sure that each training/validation/test set contains 50% benign samples and an equal amount of malware instances from each family. In this experiment, we use this small subset of the data since the PROBIT GP-LMM can be only applied on rather small datasets. In Section 6.6.6, we demonstrate that LOGISTIC GP-LMM version

[7] `http://user.informatik.uni-goettingen.de/~darp/drebin/download.html`

[8] Geinimi, FakeDoc, Kmin, Iconosys, BaseBridge, GinMaster, Opfake, Plankton, FakeInstaller, DroidKungFu.

of our model trained via *augmented variational inference* easily scales to the whole dataset containing 120,000 instances.

Since no side information is available, we only use a linear kernel and the identity matrix as components for the correlation matrix. We report on the (normalized) area under the Receiver Operating Characteristic (ROC) curve over the interval $[0, 0.1]$ and denote this performance measure by $\text{AUC}_{0.1}$. In Fig. 6.4, we show the ROC curves and in Table 6.2 the achieved $\text{AUC}_{0.1}$ and the run times.

We observe that the PROBIT GP-LMM achieves a consistent improvement in terms of $\text{AUC}_{0.1}$ over sparse PROBIT REGRESSION (by approximately 7.5 percentage points), GP CLASSIFICATION (by approximately 5 percentage points), LMM-LASSO (by approximately 8.4 percentage points). In this experiment, the LOGISTIC GP-LMM achieves a slighly less $\text{AUC}_{0.1}$ (by 0.8 percentage points) than the probit version of our model.

Furthermore, in Fig. 6.5, right, we plot the correlation of the top features of GP-LMM models and sparse PROBIT REGRESSION with population structure. We observe that both GP-LMM models lead to features that are less correlated with the malware structure.

6.6.5 Flowering time prediction from single nucleotide polymorphisms

We experiment on genotype and phenotype data consisting of 199 genetically different accessions (instances) from the model plant *Arabidopsis thaliana* (Atwell et al., 2010). The genotype of each accession comprises 216,130 single nucleotide polymorphism (SNP) features. The phenotype that we aim to predict is early or late flowering of a plant when grown at ten degrees centigrade. The original dataset contains the flowering time for each of the 199 genotypes. We split the dataset into the lower and upper 45%-quantiles of the flowering time and binarized the labels, resulting in a set of 180 accessions from which we use $n = 150$ accessions for training.

The results are reported in Table 6.3 and show that the LOGISTIC GP-LMM has a slight advantage of at least 0.7 percentage points in AUC over the competitors. In this experiment, the PROBIT GP-LMM has a slightly worse AUC than its logistic counterpart by 0.2 percentage points.

An analysis restricted to the ten SNPs with largest absolute regression weights in our models show that they lie within four well-annotated genes that all convincingly can be related to flowering, structure and growth: the gene AT2G21930 is a growth protein that is expressed during flowering, AT4G27360 is involved in microtubule

	AUC	Time
Probit GP-LMM	$84.1\% \pm 0.2$	21.02 sec
Logistic GP-LMM	$\mathbf{84.3\% \pm 0.2}$	18.31 sec
Probit Regression	$83.5\% \pm 0.2$	10.59 sec
GP Classification	$83.6\% \pm 0.2$	11.13 sec
LMM-Lasso	$79.7\% \pm 0.2$	**8.71** sec

Table 6.3: FLOWERING: Average AUC and corresponding standard deviations attained in the flowering time prediction experiment along with average training time in seconds. Both versions of our GP-LMM model perform similarly well and beat the competitors in terms of prediction performance.

	AUC	Time
Logistic GP-LMM	$\mathbf{98.0\% \pm 0.0}$	628 sec
Probit Regression	$\mathbf{98.0\% \pm 0.0}$	**250** sec
X-GPC	$96.0\% \pm 0.0$	310 sec

Table 6.4: BIG DATA: Average AUC and corresponding standard deviations attained on the full Drebin dataset containing 120,000 instances.

motor activity, AT3G48320 is a membrane protein, involved in plant structure, and AT5G28040 is a DNA binding protein that is expressed during flowering.

6.6.6 Big data experiment

In our last experiment, we show that the LOGISTIC GP-LMM model trained via *augmented variational inference* easily scales to big datasets. This is a big advantage of the logistic likelihood version of our model over the PROBIT GP-LMM, which is limited to datasets containing only a few hundred instances.

We apply LOGISTIC GP-LMM, PROBIT REGRESSION trained via mini-batch sampling and GP classification trained with the scalable inference algorithm presented in Chapter 3 (X-GPC) to the full Drebin dataset from Section 6.6.4 (where we only used a subset of 200 samples). The full dataset contains 120,000 samples. For each method, we use a mini-batch of size 2000 and use 200 inducing points. For training we use 95% of the data and for testing the remaining 5%. The hyperparameters are optimized as part of the training procedure as explained in Section 6.5. We repeat the whole procedure five times and present the attained AUCs along with one standard deviation and the training times in Table 6.4.

We find that LOGISTIC GP-LMM and PROBIT REGRESSION achieve an AUC of 98.0% and X-GPC is slightly worse by 2 percentage points. More importantly, this

experiment demonstrates the scalability of LOGISTIC GP-LMM and shows that our model can be applied to big datasets. We leave additional experiments on big genetic datasets for future research.

Chapter 7

Generalized Dynamic Topic Models

In this chapter, we propose a new approach to dynamic topic modeling. Dynamic topic models (DTMs) model the evolution of prevalent themes in literature, online media and other forms of text over time. DTMs assume that word co-occurrence statistics change continuously and therefore impose continuous stochastic process priors on their model parameters. These dynamical priors make inference much harder than in regular topic models and also limit scalability. In this chapter, we present several new results around DTMs. First, we extend the class of tractable priors from Wiener processes to the generic class of Gaussian processes. This allows us to explore topics that develop smoothly over time, that have a long-term memory or are temporally concentrated (for event detection). Second, we show how to perform scalable approximate inference in these models based on ideas around stochastic variational inference and sparse Gaussian processes. We propose an inference method that slightly deviates from the core concepts of *augmented variational inference* as presented in Chapter 2. Although we show that the novel generalized dynamic topic model is amenable to an augmentation approach, we develop a different inference approach, which is more efficient for this model. Our inference method is based on a variational Taylor expansion that approximates the softmax function. This way we can train a rich family of DTMs to massive data. Our experiments on several large-scale datasets show that our generalized model allows us to find interesting patterns that were not accessible by previous approaches.

This chapter is based on:

> P. Jähnichen*, F. Wenzel*, M. Kloft, S. Mandt (2018). "Scalable Generalized Dynamic Topic Models". In: *Conference on Artificial Intelligence and Statistics.*

Probabilistic topic models help us to organize and browse large collections of documents (Blei, 2012). Topic models have been successfully applied in information retrieval (McCallum, Corrada-Emmanuel, and Wang, 2004; Wang, McCallum, and Wei, 2007; Charlin and Zemel, 2013), computational biology (Pritchard, Stephens, and Donnelly, 2000; Gopalan et al., 2016), recommendation systems (Wang and Blei, 2011) and computer vision (Fei-Fei and Perona, 2005; Chong, Blei, and Li, 2009). Topic models assume that all words in a document were independently drawn from a finite set of probability distributions over words, termed the "topics". This way, every document is a mixture of topics. The limitation is that this approach assumes that topics are static.

Topics change over time. To provide some intuition, consider the example of the topic *"technology"* when training topic models on historical articles [1]. Restricting the corpus to articles around 1900, we find words such as *"engine"*, *"electricity"* and *"wire"* to be mainly associated with this topic. For modern articles, we may find *"devices"*, *"gates"* and *"silicon"* among the top words. In applications as this, we want to be able to associate documents with similar topic proportions with each other over large time spans. But at the same time, we want to allow topics to "modernize", meaning to dynamically adjust their vocabulary. This is achieved in dynamic topic models (DTMs) (Blei and Lafferty, 2006; Wang and McCallum, 2006; Wang, Blei, and Heckerman, 2008). DTMs model the evolution of topics as a continuous Wiener process. This dynamic prior determines how strongly topics may change their vocabulary. This way, DTMs share statistical strengths over all times, while giving the topics enough flexibility to change.

Current formulations of dynamic topic models are subject to the major limitation that they are restricted to a particular type of stochastic process for the latent topical dynamics, namely Wiener processes (i.e. Brownian motion drift priors). This formulation does not allow us to analyze long-term effects, events, or other more complicated temporal dependencies. Second, relying on the forward-backward algorithm,

*Equal contributions.

[1] Example from David Blei's tutorial slides on topic modeling, `http://www.cs.columbia.edu/~blei/talks/Blei_ICML_2012.pdf`

they lack scalability. If the data are distributed across many different time-stamps, they require a full pass through the data in every iteration. This lack of scalability may be the reason why DTMs have been much less used in large-scale scientific or industrial applications than their static counterparts. In this work, we generalize dynamic topic models in two ways: first, we extend the class of tractable priors from Wiener processes to the more general class of Gaussian processes (GPs). Second, we derive a scalable approximate Bayesian inference algorithm based on inducing points and a variational Taylor approximation. This allows us to apply our model to contemporary large text collections.

7.1 Background and related work

We connect to dynamic and correlated topic models, sparse GPs and stochastic variational inference (SVI).

Dynamic topic models. DTMs form the basis of our approach. While Blei and Lafferty (2006) originally propose a model with equidistant time slices, Wang, Blei, and Heckerman (2008) extended the approach to continuous time. Both rely on a latent Wiener process and use the forward-backward algorithm for learning, which requires full passes through the data in every iteration if the number of time stamps is comparable with the total number of documents. Wang and McCallum (2006) propose a different approach where time is an observed variable with some prior over a finite time interval. While in principle being scalable, the resulting topics are non-smooth. Finally, Bhadury et al. (2016) propose a new approach for learning in topic models based on stochastic gradient MCMC (Welling and Teh, 2011; Mandt, Hoffman, and Blei, 2016). Their approach similarly is restricted to latent Wiener processes.

Correlated and GP topic models. This class of modified static topic models breaks the independence assumptions of the per-document topic proportions. Instead, the topic proportions are jointly drawn from some prior that induces correlations (Blei and Lafferty, 2007). If this prior is a GP, this leads to the kernel topic model (Hennig et al., 2012) or Gaussian process topic model (Agovic and Banerjee, 2010). Note that both approaches assume that the topics themselves are static and

only the topic proportions change. In contrast, we treat the proportions as independent and identically distributed (iid) and impose dynamics on the topics themselves. None of these models have been formulated in a scalable manner.

7.2 Generalized dynamic topic models

Dynamic topic models are mixed-membership bag-of-words models, which allow their mixture components—the topics—to drift over time. This allows to dynamically fade-in new words and fade-out old words, which lose their semantic significance in a topic. In the classic DTM this continuity is achieved by imposing a Wiener process prior on the topic matrices (Blei and Lafferty, 2006; Wang, Blei, and Heckerman, 2008) (see also Bamler and Mandt, 2017 for a related approach for word embeddings).

In this work, we propose GPs as priors on the topic matrices. Since the Wiener process is a specific type of GP, our approach is a strict generalization of dynamic topic models but covers a much richer class of dynamics.

7.2.1 Generalized DTMs

For what follows, we borrow notation from the topic modeling literature (Blei, Ng, and Jordan, 2003). We assume that we observe a corpus of D documents, each of which is associated with a time stamp τ_{t_d} with index $t_d \in \{1, \dots, T\}$. For a simpler notation we denote the number of words in a document as N. For a given document d (which is associated with one time index t), let $w_{d1}, \cdots, w_{dN}$ be the words it contains, $\boldsymbol{\theta}_d$ be a K-vector of topic proportions and z_{dn} the assignment of word w_{dn} to a topic. The model consists of K time dynamic topics whereby $\boldsymbol{\beta}_{k\cdot t}$ denotes a topic's V-dimensional distribution over the vocabulary at time index t.

Our model exhibits the following joint distribution:

$$p(w, z, \theta, \beta) = p(\beta) \prod_{t=1}^{T} \prod_{k=1}^{K} p(w_t, z_t, \theta | \sigma(\boldsymbol{\beta}_{k\cdot t})), \tag{7.1}$$

where each document d is associated with one time index t_d. The function $\sigma(\cdot)$ is the softmax function, which normalizes the topic $\boldsymbol{\beta}_{k\cdot t}$ over the vocabulary indices. The joint distribution of the words, topic assignments and topic proportions conditioned on the topic trajectories at time index t,

$$p(w_t, z_t, \theta | \beta_{\cdot\cdot t}) = \prod_{d:t_d=t} p(\boldsymbol{\theta_d}) \prod_n p(w_{dn} | \sigma(\boldsymbol{\beta}_{z_{dn}\cdot t})) p(z_{dn} | \boldsymbol{\theta_d}),$$

is just a regular LDA model (at time index t), with

$$\begin{aligned} w_{dn}|\boldsymbol{\beta}_{z_{dn}\cdot t} &\sim \text{Cat}(\sigma(\boldsymbol{\beta}_{z_{dn}\cdot t})) \\ z_{dn}|\boldsymbol{\theta_d} &\sim \text{Cat}(\boldsymbol{\theta_d}) \\ \boldsymbol{\theta_d} &\sim \text{Dir}(\alpha), \end{aligned}$$

where Cat(.) denotes the categorical distribution and Dir(.) the Dirichlet distribution. The graphical model is shown in Fig. 7.1.

The distinctive feature of dynamic topic models is their dynamic prior $p(\beta)$. In our model, each of the V words out of K topics is a latent function over time, drawn from a GP with kernel function κ. This GP is observed at time indices $t = 1, \cdots, T$ associated with the time stamps $\tau_1, \cdots, \tau_T$ and can thus be described as a T-dimensional multivariate Gaussian distribution[2]

$$\beta_{kw\cdot} \sim \text{GP}(0, \kappa) \Rightarrow \boldsymbol{\beta}_{kw,1:T} \sim \mathcal{N}(0, K_{TT}), \tag{7.2}$$

where the entries of the kernel matrix are $K_{t,t'} = \kappa(\tau, \tau')$. The kernel function κ measures the similarity between two time points τ and τ'. Using a Wiener kernel function in our model results in the classic DTM of Wang, Blei, and Heckerman (2008). However, due to the model's flexibility, we can model *any* stochastic process that falls into the class of GPs by simply altering the covariance function κ. As an aside, this setup not only covers the dynamic setup but also allows for incorporating other types of meta data as e.g. spatial modeling if the text documents are associated with location coordinates.

In this work, we focus on the time-specific setup. In more detail, we consider several different kernels commonly used for time-series modeling (Roberts et al., 2012).

- **Wiener kernels**, $\kappa_{\text{Wie}}(\tau, \tau') = \sigma_\beta^2 \min(\tau, \tau')$. Using a Wiener kernel (Brownian motion kernel) in our model recovers the typical DTM setup. This serves as our baseline.
- **Ornstein-Uhlenbeck kernels**, $\kappa_{\text{OU}}(\tau, \tau') = \sigma_\beta^2 \exp\left(-\frac{|\tau - \tau'|}{l}\right)$. The Ornstein-Uhlenbeck (OU) process is essentially a Wiener process in the presence of a mean-reverting force, which pulls the process state back to its mean and thus acts like a regularizer. An effect of this is that topics may die-off and other topics may dynamically emerge (using a zero-mean process). As we show in our experiments, this leads to temporally localized changes in topics.

[2]We call attention to the slight overloading of notation: a plain K always is the number of topics, using subscripts or a tilde it denotes a kernel/covariance matrix.

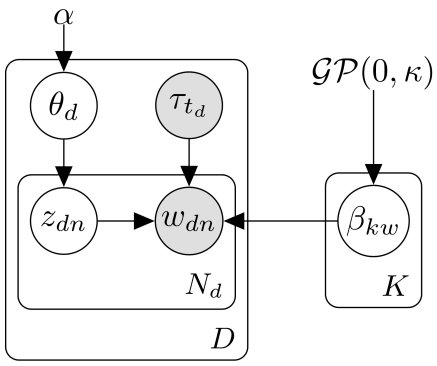

Figure 7.1: The generalized dynamic topic model.

- **RBF kernels**, $\kappa_{\text{RBF}}(\tau,\tau') = \sigma_\beta^2 \exp\left(-\frac{(\tau-\tau')^2}{2l^2}\right)$. RBF kernels, also known as squared exponential kernels, have the property that the resulting trajectories are smoother compared to Wiener kernels. The resulting prior functions are infinitely often differentiable. The exponential decay of the temporal correlations leads to memory effects that can be parameterized by the kernel's length scale l. With a suitable chosen l this allows for temporally localized topics.
- **Cauchy Kernels**, $\kappa_{\text{Cau}}(\tau,\tau') = \sigma_\beta^2 \left(1 + \frac{(\tau-\tau')^2}{l^2}\right)^{-1}$. Cauchy kernels are constructed similarly as RBF kernels, but instead of using the Gaussian density one uses a Cauchy density. This kernel has long-range memory, which means that temporal correlations decay not exponentially but polynomially, which in some cases is more realistic.

Note that *any* additive or multiplicative combination of covariance functions again results in a valid covariance function and so can similarly be used. This adds considerably to the flexibility of the proposed prior.

We again stress that all these kernels use the same inference algorithm. The problem is that a naive implementation would scale cubically in the number of time stamps. We, therefore, propose a more efficient version based on the concept of sparse GPs.

7.2.2 A side note on augmenting the model

In this section, we briefly sketch the possibility of obtaining a conditionally conjugate augmentation of our model and connect the model to our general *augmented variational inference* framework. However, we argue that a direct *augmented variational inference* approach is not practical for this model and we develop an alternative scalable inference method in Section 7.3.

In the previous section, we saw that, conditioned on the topic matrices β, our model corresponds to a standard LDA model, which is a conditionally conjugate model (Blei, Ng, and Jordan, 2003). Therefore, the variables w, z and θ in our model are conditionally conjugate and the corresponding complete conditional distributions are computed in closed form.

In contrast, the complete conditional distribution of the topic matrix $\beta_{k\cdot\cdot}$ is more challenging. For each time index t, the word probability vector $\boldsymbol{\beta}_{k\cdot t}$ is mapped through a softmax to form a categorical likelihood

$$p(w_{dn}|\boldsymbol{\beta}_{z_{dn}\cdot t}) = \text{Cat}(w_{dn}|\sigma(\boldsymbol{\beta}_{z_{dn}\cdot t})).$$

Hence, computing the complete conditionals of β is essentially a multi-class GP classification inference problem. In Chapter 4, we have discussed an *augmented variational inference* approach to multi-class GP classification. In principle, a similar strategy could be applied here by replacing the softmax likelihood by our novel *logistic-softmax* (see Eq. 4.5). Consequently, a similar augmentation approach as presented in Chapter 4 could be applied and would turn the generalized dynamic topic model into a conditionally conjugate model.

Although this is an interesting idea, this approach has the disadvantage that an individual multi-class GP augmentation would be necessary for each topic k. Additionally, each topic consists of V words and the vocabulary size often goes into the thousands. An augmentation of the corresponding multi-class GP classification models with a thousand classes would be not feasible.

We, therefore, follow a different approach and in the next section, we develop an alternative inference method, where we use the original softmax likelihood and introduce only T additional auxiliary variables ζ_{tk} for each topic k, based on a variational Taylor approximation.

7.3 Inference

We first propose a tractable lower bound on the likelihood employing a Taylor expansion. We then show how we can decompose the ELBO into a part that is equivalent to LDA and into another part that contains the GP prior and therefore is more complex. We list all modified updates on the local and global parameters, with detailed calculations given in Appendix E.2.

Dealing with the intractable likelihood. As discussed in the previous section, our model, conditioned on the topic trajectories β, is conditionally conjugate and the variational updates can be computed by the standard CAVI approach as outlined in Section 2.2. Hence, we first consider the problematic likelihood term $p(w|\beta_k)$ and propose a tractable lower bound by introducing a set of auxiliary variables. We follow a similar approach as Blei and Lafferty, 2009 and compute the first order Taylor approximation of the logarithm around an arbitrary location parameter $\zeta_{kt} > 0$,

$$\begin{aligned}
\log p(w_{dn}|z_{dn} = k, \beta, t_d) &= \log \mathrm{Cat}\left(w_{dn}|\sigma(\boldsymbol{\beta}_{k\cdot t_d})\right) \\
&= \beta_{kw_{dn}t_d} - \log \sum_w \exp(\beta_{kwt_d}) \\
&\geq \beta_{kw_{dn}t_d} - \zeta_{kt_d}^{-1} \sum_w \exp(\beta_{kwt_d}) - \log(\zeta_{kt_d}) + 1 \\
&=: \tilde{p}(w_{dn}|z_{dn} = k, \beta, t_d, \zeta_{kt_d}). \qquad (7.3)
\end{aligned}$$

Next we show how to use this bound to obtain a tractable variational objective. In each iteration of our inference algorithm, we optimize the location parameter of the Taylor expansion to achieve the tightest possible bound on the true marginal likelihood.

Sparse GP approximation. We scale our model to big datasets by approximating the latent GPs β_{kw} by *sparse GPs* building on *inducing points*. For each GP β_{kw}, which is defined on T time stamps, we introduce $\hat{T}$ inducing points u_{kw} and connect the GP values with the inducing points via the joint prior distribution $p(\beta_{kw}, u_{kw})$ given in Section 1.2. Let K_{TT} be the kernel evaluated at all training points, $K_{\hat{T}\hat{T}}$ the kernel evaluated at inducing points, and $K_{T\hat{T}}$ and $K_{\hat{T}T}$ be kernels evaluated in-between these sets of points.

Variational bound. We follow a variational structured mean-field approach (Wainwright and Jordan, 2007) and impose the following variational distributions on the latent variables,

$$\begin{aligned}
q(\boldsymbol{\theta}_d) &= \mathrm{Dir}(\boldsymbol{\lambda}_d) \\
q(z_{dn}) &= \mathrm{Cat}(\boldsymbol{\phi}_{dn}) \\
q(\boldsymbol{u}_{kw\cdot}) &= \mathcal{N}(\boldsymbol{\mu}_{kw}, \Sigma_{kw}).
\end{aligned}$$

Eq. 7.3 gives rise to the following tractable lower bound of the marginal likelihood:

$$\begin{aligned}\mathcal{L}(\lambda,\phi,\mu,\Sigma,\zeta) &= \mathbb{E}_{q(\theta)q(z)p(\beta|u)q(u)}[\log\tilde{p}(w|\beta,z,\zeta)p(z|\theta)p(\theta)p(u)] + H(q) \qquad (7.4)\\ &= \underbrace{\mathbb{E}_{p(\beta|u)q(u)}[\log\tilde{p}(w|\beta,z,\zeta)]}_{\mathcal{L}_1} + \underbrace{\mathbb{E}_{q(\theta)q(z)}[\log p(z|\theta)p(\theta)p(u)]}_{\mathcal{L}_2} + \underbrace{H(q)}_{\text{Entropy}},\end{aligned}$$

where $H(q) = -\mathbb{E}_{q(\theta,z,u)}[\log q(\theta,z,u)]$ is the entropy of the variational distribution. The term $\mathcal{L}_2$ and the entropy term can be computed similarly as in standard LDA using standard results. Details are provided in Appendix E.1.

We now show that also the first term $\mathcal{L}_1$ can be computed in closed form. We first compute the expectation w.r.t. $p(\beta|u)$ and obtain

$$\begin{aligned}\tilde{p}(w_{dn}|z_{dn}=k,u,t_d,\zeta_{kt_d}) &:= \mathbb{E}_{p(\boldsymbol{\beta}_{k\cdot t_d}|u)}[\tilde{p}(w_{dn}|z_{dn}=k,\beta,t_d,\zeta_{kt_d}] \qquad (7.5)\\ &= \boldsymbol{K}_{t_d\hat{T}}K^{-1}_{\hat{T}\hat{T}}u_k w_{dn} - \zeta^{-1}_{kt_d}\sum_w \exp\left(\boldsymbol{K}_{t_d\hat{T}}K^{-1}_{\hat{T}\hat{T}}u_{kwt_d} + \frac{\tilde{K}_{t_dt_d}}{2}\right) - \log(\zeta_{kt_d}) + 1,\end{aligned}$$

where u_k is a $\hat{T}\times V$ matrix and $\boldsymbol{K}_{t_d\hat{T}}$ is the t_d-th row of $\boldsymbol{K}_{t\hat{T}}$. The final expression is obtained by computing the expectation w.r.t. variational distribution $q(u)$, which gives

$$\begin{aligned}\mathcal{L}_1 = \sum_{t,k,w}\sum_{d:t_d=t} n_{dw}\phi_{dwk}\Bigg\{ & m_{kwt} - \log(\zeta_{kt}) + 1\\ & -\zeta^{-1}_{kt}\sum_{w'}\exp\left(m_{kw't} + \frac{1}{2}(\Lambda_{kw't} + \tilde{K}_{tt})\right)\Bigg\},\end{aligned}$$

with

$$\begin{aligned}m_{kwt} &= \boldsymbol{K}_{t\hat{T}}K^{-1}_{\hat{T}\hat{T}}\boldsymbol{\mu}_{kw}\\ \Lambda_{kwt} &= \boldsymbol{K}_{t\hat{T}}K^{-1}_{\hat{T}\hat{T}}\Sigma_{kw}K^{-1}_{\hat{T}\hat{T}}\boldsymbol{K}_{\hat{T}t}.\end{aligned}$$

The variational objective $\mathcal{L}$ is optimized using SVI (Hoffman et al., 2013), i.e. for global variational parameters, we follow noisy natural gradients based on mini-batches. Local variational parameter updates are updated using coordinate ascent and are similar to those in the standard dynamic topic model (Wang, Blei, and Heckerman, 2008). Further details are provided in Appendix E.1.

Global updates. We consider the Gaussian distributions $q(\boldsymbol{u}_{kw\cdot})$ in *natural parameterization*, i.e. using the parameters $\boldsymbol{\eta}^{(1)}_{kw} = \Sigma^{-1}_{kw}\boldsymbol{\mu}_{kw}$ and $\eta^{(2)}_{kw} = -\frac{1}{2}\Sigma^{-1}_{kw}$, where $\boldsymbol{\mu}_{kw}$ are the Gaussian means and Σ_{kw} the covariances. In SVI, we update these global

parameters using stochastic estimates of the natural gradient and it turns out that in this case, using natural parameters result in simpler and more effective updates.

More specifically, for a Gaussian distribution, properties of the Fisher information matrix expose the simplification that the *natural gradient* w.r.t. the natural parameters can be expressed in terms of the *Euclidean gradient* w.r.t. the canonical parameters (i.e. mean and covariance). Namely, in general, it holds for objectives $\mathcal{F}$ that depend on a Gaussian distribution that

$$\hat{\nabla}_{(\boldsymbol{\eta}_1,\eta_2)}\mathcal{F}(\eta) = \big(\nabla_{\boldsymbol{\mu}}\mathcal{F}(\eta) - 2\nabla_{\Sigma}\mathcal{F}(\eta)\boldsymbol{\mu},\ \nabla_{\Sigma}\mathcal{F}(\eta)\big), \tag{7.6}$$

where $\hat{\nabla}$ denotes the natural gradient and ∇ the Euclidean gradient. Applying (7.6) to the variational objective (7.4), we obtain

$$\begin{aligned}\hat{\nabla}_{\boldsymbol{\eta}^{(1)}_{kw}}\mathcal{L} &= \boldsymbol{\Xi_{kw}} + \boldsymbol{B_{kw}} \circ (\boldsymbol{m_{kw}} - 1) - \boldsymbol{\eta}^{(1)}_{kw},\\ \hat{\nabla}_{\eta^{(2)}_{kw}}\mathcal{L} &= -\frac{1}{2}K^{-1}_{\hat{T}\hat{T}} - \frac{1}{2}C_{kw} - \eta^{(2)}_{kw}.\end{aligned} \tag{7.7}$$

We used the following abbreviations:

$$\begin{aligned}\boldsymbol{\Xi_{kw}} &= K^{-1}_{\hat{T}\hat{T}}\sum_t \sum_{d:t_d=t} \boldsymbol{K}_{\hat{T}t} n_{dw}\phi_{dwk},\\ \boldsymbol{B_{kw}} &= \sum_t \sum_{d:t_d=t} \zeta^{-1}_{kt} n_{dw}\phi_{dwk} \exp\left(m_{kwt} + \frac{\Lambda_{kwt} + \tilde{K}_{tt}}{2}\right) K^{-1}_{\hat{T}\hat{T}}\boldsymbol{K}_{\hat{T}t},\\ C_{kw} &= \boldsymbol{B_{kw}}\boldsymbol{K}_{t\hat{T}}K^{-1}_{\hat{T}\hat{T}}.\end{aligned}$$

Above, $\circ$ denotes the Hadamard product. Details are provided in Appendix E.1. Iterating through those updates completes the algorithm.

Taylor expansion location parameter updates. Setting the derivative of $\mathcal{L}$ w.r.t. ζ_{kt} to zero and solving for ζ_{kt} gives the update

$$\zeta_{kt} = \sum_v \exp\left(m_{kvt} + \frac{1}{2}(\Lambda_{kvt} + \tilde{K}_{tt})\right).$$

7.4 Experiments

We evaluate our method on three time-stamped text corpora. Compared to standard DTMs with Wiener kernels, we find that incorporating other dynamic priors may lead to improved predictive likelihoods and perplexity on hold-out data. Using different

kernel functions within our framework, we find new insights in the data that could not be found using the classic DTM of Wang, Blei, and Heckerman (2008), which uses the Wiener kernel.

By making use of the greater flexibility that comes with general GPs, we show how to extend and enhance an analysis. For instance, by using the *OU kernel*, we introduce a mean-reverting force that quickly "draws" word probabilities towards zero, resulting in topics that are consistent and constrained in time and more sensitive to changes. Further, in situations in which the classic approach collapses most of the probability mass to single words per time stamp, we compensate by using *Cauchy kernels*, which place a smooth filter on word probabilities onto the topic over time. On the other hand, more fine-grained temporal dynamics can be captured by *RBF kernels*, due to their short-range memory.

Data and preprocessing.

1. We use the "The New York Times Annotated Corpus" (NYT) (Sandhaus, 2008), which consists of over 1.8 million articles published between 1987 and 2007 with $T = 7475$ unique time stamps. We subsample 100,000 documents.

2. We use the *NIPS* dataset that contains 2711 papers from the NIPS conferences between 1987 and 2006[3] resulting in $T = 19$ time stamps.

3. We use the "State of the Union" (SoU) addresses of U.S. presidents, which span more than two centuries, resulting in $T = 224$ different time stamps [4]. We increase the number of documents to 4428 by treating every chunk of ten paragraphs in a speech as a separate document.

For preprocessing, we filtered the raw data using a standard stop word list. After collecting word statistics, we remove words that appear less than 25 times across a whole corpus. We further shrink the vocabulary by removing words whose tf-idf score is less than a certain threshold, resulting in dictionaries of reasonable size (see Appendix E.3). After this step, we remove documents with effective lengths less than ten word occurrences. We initialize our models by randomly selecting K documents for any given time stamp and setting probabilities in topic k of occurring words proportional to their frequencies in the k-th document.

[3] `http://www.datalab.uci.edu/author-topic/NIPs.htm`

[4] `http://www.presidency.ucsb.edu/sou.php`

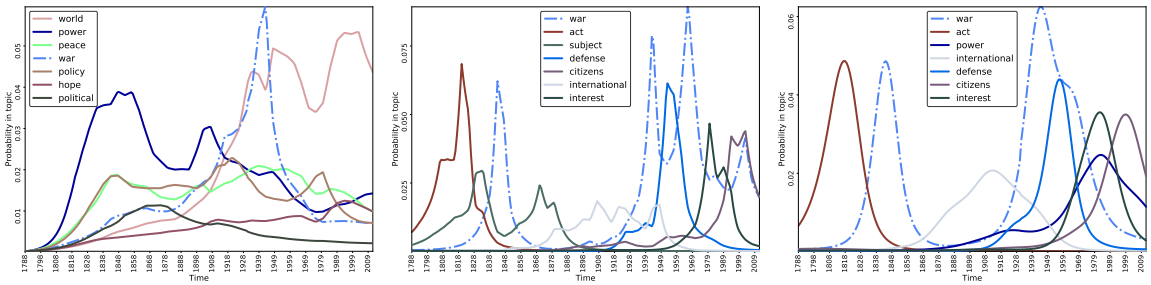

Figure 7.2: *SoU*: Learned word trajectories of the *"war"* topic using the Wiener kernel (left), OU kernel (middle) and Cauchy kernel (right). The Cauchy kernel provides smoother trajectories yet the OU kernel is able to provide a better resolution in time. Both outperform the baseline in terms of perplexity.

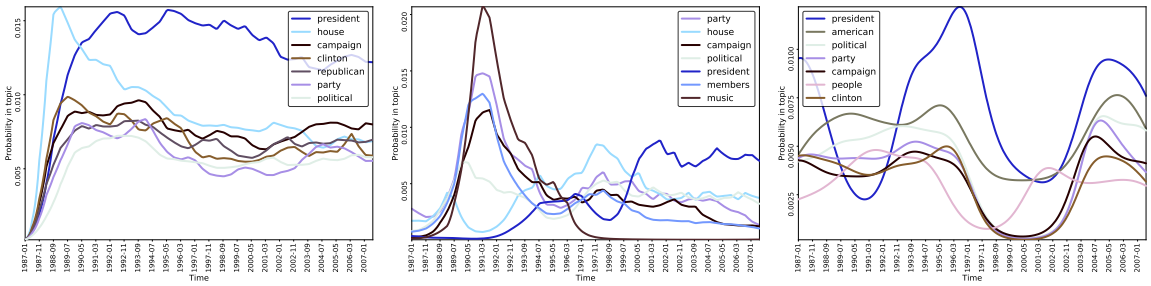

Figure 7.3: *NYT*: Learned word trajectories of the *"election campaign"* topic using the Wiener kernel (left), OU kernel (middle) and Cauchy kernel (right), which results in the smoothest curves.

Hyperparameters. In our experiments, we select the hyperparameters via grid search but we remark that they could also be directly learned in our inference scheme using the approximate empirical Bayes approach outlined in Section 2.2.

Qualitative results. We now discuss the qualitative results obtained from applying our model on all three corpora. For certain example topics, we plot and discuss the probabilities of the most important words in this topic over time. As a general tendency, we find that our proposed Ornstein-Uhlenbeck and Cauchy kernels outperform the standard Wiener kernel in terms of interpretability and in terms of usefulness for detecting events.

SoU. We fit our generalized DTM with different kernels to the state-of-the-union corpus and for each kernel select the topic that includes *"war"* and *"peace"* as it top words. Fig. 7.2 shows the word probabilities within this topic over time for all three considered kernels. The Wiener kernel is able to find a semantically coherent

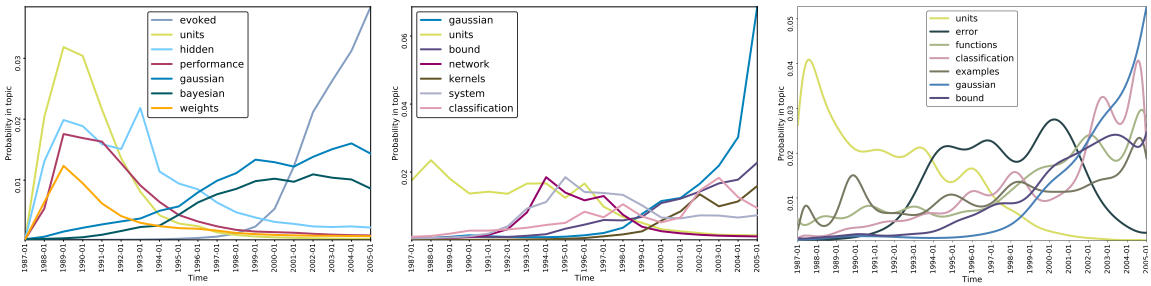

Figure 7.4: *NIPS*: Learned word trajectories of the *"function approximation"* topic using the Wiener kernel (left), OU kernel (middle) and Cauchy kernel (right). All three approaches identify terms that gain or loose importance within the topic over time. Since the Cauchy kernel shares statistical strength over a broader time horizon, its word trajectories are smoother.

word distribution for this topic. We observe a relatively high probability of the term *"war"* over the whole time span with a sharp peak around 1939 (World War II). Using the Cauchy kernel, we are able to gain a better resolution of the dynamics for the importance of this term. We observe two separate high-probability periods of the word *"war"*. One is matching the time of the American-Mexican war 1846-1848, the other one the World Wars and Vietnam war. We attribute this finding to the fact that the Cauchy kernel shares more statistical strength over time due to its long-term memory property.

While this model already provides a better insight into active time periods of the topic, additionally introducing a mean-reverting force via the OU kernel provides a mean to "super-resolve" topic activity quite accurately to certain events. We observe high probability for the term *"war"* again around 1848, a small plateau in the 1910s (World War I) rising to a high value in 1939 (World War II) and the 1960s (Vietnam war). We even observe a small bump in the beginning and through the 1980s (possibly the war in Afghanistan) and another peak in the mid 2000s (second Afghanistan war). Additionally, when looking at the words with highest probability at these times, we observe that the model is able to place probability mass on terms relating to the different wars, e.g. *"texas"* for the American-Mexican war (which was fought over Texas) or *"attack"* and *"japanese"* in 1942 (where the attack on Pearl Harbor took place). Based on these findings, the Ornstein-Uhlenbeck kernel seemed most appropriate for this task.

NYT. Another interesting use case scenario is the analysis of news texts. One of the topics identified when analyzing the New York Times corpus deals with presidential election campaigns. Fig. 7.3 shows probability trajectories of terms in that

topic for the Wiener (left), Ornstein-Uhlenbeck (middle) and Cauchy (right) kernels, respectively. We observe that the baseline model (Wiener kernel) is able to capture meaningful terms. Going beyond this, the OU kernel arguably reflects the election campaigns in 1992 and 1996. The Cauchy kernel results in even smoother trajectories. These findings, however, deserve a more thorough investigation and interpretation. Nevertheless, this example shows that different kernels reveal qualitatively different phenomena.

NIPS. Applying generalized DTMs on the *NIPS* corpus allows us to track trends in machine learning over the last 20 years. We present probability trajectories of a topic related to classification and function approximation. Again, we show results for Wiener, OU and Cauchy kernels (Fig. 7.4, left to right). We observe from the baseline model that neural networks gained attention in the late 1980s and early 1990s. However, the excitement subsided in the late 1990s and Bayesian methods were on the rise (our dataset is not recent enough to detect the uptrend of neural networks in the last ten years). While the Wiener kernel models overall development, the Ornstein-Uhlenbeck process is able to better react to small scale changes, resulting in a more realistic representation of term development. Additionally, it finds more general terms, such as *"network"*, *"classification"* and *"system"*. Using the Cauchy kernel with its long-term memory prevents from placing large probability mass on the rapidly rising term *"gaussian"*. The Cauchy kernel is also able to identify more general terms.

Quantitative results. We show that using our approach not only leads to interesting dynamic topics but also generalizes better to unseen data. We use all documents associated with time stamps T_{train} as training set and analyze the predictive hold-out likelihoods on remaining documents (associated with the time stamps $T_{\text{test}} = T \setminus T_{\text{train}}$). We experiment on the *NYT* dataset and randomly select T_{train} to hold 85% of the unique time stamps.

Data	cDTM (baseline)	gDTM OU	gDTM Cauchy	gDTM RBF
NYT	1.42323	**1.42073**	1.42129	1.42374
NIPS	1.4931	1.48149	**1.48105**	1.4821
SoU	1.46854	1.45594	**1.45575**	1.46023

Table 7.1: Per-word predictive perplexities (lower numbers are better). We constantly outperform the baseline on all datasets.

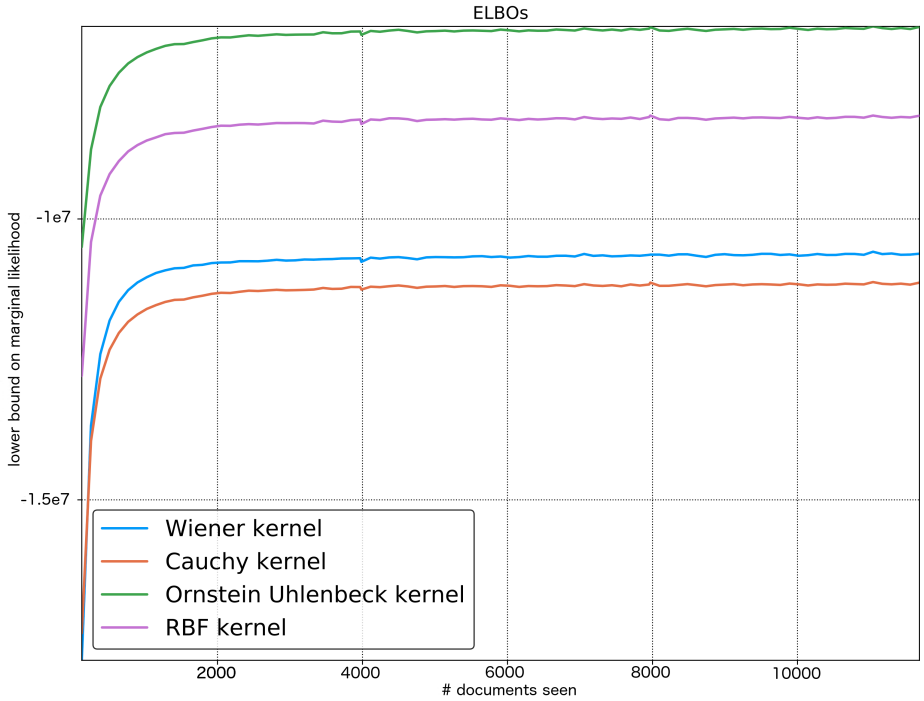

Figure 7.5: *SoU*: Evidence lower bound against the number of documents seen. On all used kernels, the objective function converges to an optimum.

Table 7.1 shows that our method outperforms the baseline in terms of per-word predictive perplexity (see e.g. Blei and Lafferty, 2007). We observe that the perplexity on both the *NIPS* and *SoU* dataset is best when the dynamics are modeled by a GP with Cauchy kernel while the *NYT* dataset is best captured by a OU kernel. This shows again the advantage of using our approach over the state of the art. Having the flexibility of modeling the dynamics by a GP we can account for the different dynamics that may underlie different datasets. Additionally, Fig. 7.5 shows the ELBO objective function when fitting a model to the *SoU* dataset, eventually reaching an optimum. Results on the different datasets were similar.

Remarks. We argue that as common in probabilistic modeling, the prior should not be chosen based on predictive likelihood alone. Instead, a prior is a modeling choice that helps reveal the effects that one searches for. Depending on the problem at hand, a practitioner would choose the suitable kernel, be it the Wiener kernel, Ornstein-Uhlenbeck kernel, RBF kernel or Cauchy kernel. The Ornstein-Uhlenbeck kernel has the favorable property of localizing topics in time, which may be a promising tool for event detection. However, if the length scale is too small, topics change their word distributions at a frequency, which is too high in which case the results

are less interpretable. On the other hand, the RBF kernel (and even more so the Cauchy kernel) has long-time memory and is generally more data efficient, which has advantages if the dataset is small. Ultimately, many other kernels may be designed for different purposes.

Chapter 8

Conclusions and Outlook

In this thesis we have studied latent Gaussian process (GP) models. GPs provide a flexible approach to build expressive probabilistic models. They allow for incorporating prior knowledge into the model via the choice of appropriate kernel functions, they become more expressive as the number of training data increases and they lead to well calibrated uncertainty estimates. However, inference in latent GP models is challenging and existing inference methods are often slow and unstable.

Summary of results

The main contribution of this thesis was to develop a new framework for performing efficient inference in latent GP models. Our approach is based on an auxiliary variable augmentation that renders the original intractable GP model conditionally conjugate. We laid out the principles of the *augmented variational inference* approach in the first part of this thesis.

The second part of this thesis examined five latent GP models and developed new inference algorithms for each model.

In Chapter 3, we studied the well known Gaussian process classification model. We prosed a new *augmented variational inference* approach and made it more practical by making inference faster up to two orders of magnitude.

Chapter 4 proposed a new GP multi-class classification model based on the novel logistic-softmax likelihood. The new likelihood has two advantageous: it allows for an efficient auxiliary variable augmentation and leads to better uncertainty calibration than previous scalable GP multi-class classification methods.

In Chapter 5, we studied a Bayesian version of the classic support vector machine (SVM). The Bayesian SVM combines the benefits of the standard frequentist SVM (e.g. robustness against outliers) with advantages of a Bayesian formulation (e.g.

uncertainty quantification). While previous methods for the Bayesian SVM were restricted to rather small datasets, our method enables the application of the Bayesian nonlinear SVM to large real-world datasets containing millions of samples.

Chapter 6 was concerned with sparse feature selection in the setting of binary classification where the data show spurious correlations, e.g. due to confounding. We proposed the *sparse Gaussian process linear mixed model*. We discussed two different inference methods: an expectation propagation based method for the Probit likelihood version and an *augmented variational inference* method for the logistic likelihood version of our model, which scales to big datasets. We demonstrated that our method is able to disambiguate between sparse linear effects and correlated Gaussian noise and thereby explains away spurious correlations due to confounding. We showed empirically that our model selects features which show less correlation with the confounder and which are, therefore, closer to the true underlying sparsity pattern.

In Chapter 7, we developed a new latent GP model for language modeling. The *generalized dynamic topic model* allows for dynamic topic modeling with a broad class of dynamic priors. In particular, we generalized dynamic topic models from Brownian motion priors to arbitrary Gaussian process priors. Our inference method easily scales to very large text collections. We showed in the experiments that our approach leads to better predictive likelihoods on hold-out documents and to interesting new qualitative findings such as temporally localized topics and topics that display long-range temporal dependencies.

Future Work

The thesis has shown that the proposed *augmented variational inference* framework makes the exploration of new latent Gaussian process models more convenient. In practice, it makes a big difference if a researcher has to wait one minute or a hundred minutes for the inference procedure to converge. We hope that the results of this thesis will pave the way to investigations of novel interesting models and leads to new ideas for efficient inference methods.

Future work may aim at extending the *augmented variational inference* framework to new interesting models. In particular, one interesting research direction would be to investigate Bayesian neural networks (BNNs). Inference in BNNs is a challenging problem and there has been only limited success in developing efficient methods so far. Exchanging the common softmax link functions with the proposed logistic-softmax from Chapter 4 may lead to a conditionally conjugate augmentation approach for BNNs. Typically, Gaussian priors are used for the weights of the network. In the

augmented model the posterior of the weights would be also Gaussian and given in closed form. This might lead to an efficient inference algorithm.

As another research direction, there are many interesting possibilities to further investigate the proposed latent Gaussian process models. For instance, a promising improvement of the *generalized dynamic topic model* from Chapter 7 would be to extend the dynamic topics from the time domain to other domains such as to the geo-spatial domain. This could lead to interesting insights on datasets where the text is equipped with location information.

The feature selection approach from Chapter 6 is focused on finding a causal *linear* fixed effect. A fruitful extension to this model might be the incorporation of *non-linear* effects. For instance, it would be interesting to replace the linear term in our model with a deep neural network equipped with some sparsity inducing regularizer. This approach could lead to a non-linear feature selection method that can deal with confounding effects.

Finally, a more fundamental line of research is to establish a framework that would fully automize the *augmented variational inference* procedure. Although in this thesis we have provided some guidelines on how to find a suitable augmentation, it is still a complex task that has to be done manually for each model. We plan to develop a framework that, given the latent GP model, *automatically* provides a suitable augmentation and outputs the variational inference updates. First results suggest that such a procedure can be developed for a large class of likelihood functions—namely, for models where the likelihood is based on a positive radial function (Schoenberg, 1938). In these models, the distribution of a suitable auxiliary variable that renders the model conditionally conjugate can be constructed automatically. Such a fully automized procedure would be a great tool for practitioners who want to experiment with new latent GP models and need to obtain fast inference results on large datasets.

Appendix A

Gaussian process classification

A.1 Variational bound

We provide details of the derivation of the variational bound (3.6) which is defined as

$$\mathcal{L}(\boldsymbol{c}, \boldsymbol{\mu}, \Sigma) = \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{f})] - \mathrm{KL}\left(q(\boldsymbol{u}, \boldsymbol{\omega})||p(\boldsymbol{u}, \boldsymbol{\omega})\right),$$

and the family of variational distributions is

$$q(\boldsymbol{u}, \boldsymbol{\omega}) = q(\boldsymbol{u}) \prod_i q(\omega_i) = \mathcal{N}(\boldsymbol{u}|\boldsymbol{\mu}, \Sigma) \prod_i \mathrm{PG}(\omega_i|1, c_i).$$

The likelihood term simplifies to

$$\begin{aligned}
\mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})}[\log p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{f})] &\overset{\mathsf{c}}{=} \frac{1}{2}\mathbb{E}_{p(f|u)}\left[\boldsymbol{y}^\top \boldsymbol{f} - \boldsymbol{f}^\top \Omega \boldsymbol{f}\right] \\
&= \frac{1}{2}\left(\boldsymbol{y}^\top K_{nm} K_{mm}^{-1} \boldsymbol{u} - \mathrm{tr}(\Omega \widetilde{K}) - \boldsymbol{u}^\top K_{mm}^{-1} K_{mn} \Omega K_{nm} K_{mm}^{-1} \boldsymbol{u}\right).
\end{aligned}$$

Computing the expectations w.r.t. to variational distributions gives

$$\begin{aligned}
&\mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})q(\boldsymbol{u})q(\boldsymbol{\omega})}[\log p(\boldsymbol{y}|\boldsymbol{\omega}, \boldsymbol{f})] \\
&\quad\overset{\mathsf{c}}{=} \frac{1}{2}\mathbb{E}_{q(\boldsymbol{u})q(\boldsymbol{\omega})}\left[\boldsymbol{y}^\top K_{nm} K_{mm}^{-1} \boldsymbol{u} - \mathrm{tr}(\Omega \widetilde{K}) - \boldsymbol{u}^\top K_{mm}^{-1} K_{mn} \Omega K_{nm} K_{mm}^{-1} \boldsymbol{u}\right] \\
&\quad= \frac{1}{2}\mathbb{E}_{q(\boldsymbol{u})}\left[\boldsymbol{y}^\top K_{nm} K_{mm}^{-1} \boldsymbol{u} - \mathrm{tr}(\Theta \widetilde{K}) - \boldsymbol{u}^\top K_{mm}^{-1} K_{mn} \Theta K_{nm} K_{mm}^{-1} \boldsymbol{u}\right] \\
&\quad= \frac{1}{2}\Big[\boldsymbol{y}^\top K_{nm} K_{mm}^{-1} \boldsymbol{\mu} - \mathrm{tr}(\Theta \widetilde{K}) - \mathrm{tr}(K_{mm}^{-1} K_{mn} \Theta K_{nm} K_{mm}^{-1} \Sigma) \\
&\qquad - \boldsymbol{\mu}^\top K_{mm}^{-1} K_{mn} \Theta K_{nm} K_{mm}^{-1} \boldsymbol{\mu}\Big] \\
&\quad= \frac{1}{2}\sum_i \left(y_i \boldsymbol{\kappa}_i \boldsymbol{\mu} - \theta_i \widetilde{K}_{ii} - \theta_i \boldsymbol{\kappa}_i \Sigma \boldsymbol{\kappa}_i^\top - \theta_i \boldsymbol{\mu}^\top \boldsymbol{\kappa}_i^\top \boldsymbol{\kappa}_i \boldsymbol{\mu}\right),
\end{aligned}$$

where $\theta_i = \mathbb{E}_{p(\omega_i)}[\omega_i] = \frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)$, $\Theta = \mathrm{diag}(\boldsymbol{\theta})$ and $\boldsymbol{\kappa}_i = K_{im}K_{mm}^{-1}$.

The Kullback-Leibler divergence between the Gaussian distributions $q(\boldsymbol{u})$ and $p(\boldsymbol{u})$ is easily computed

$$\mathrm{KL}(q(\boldsymbol{u}))||p(\boldsymbol{u})) \stackrel{\mathsf{c}}{=} \frac{1}{2}\left(\mathrm{tr}\left(K_{mm}^{-1}\Sigma\right) + \boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} - \log|\Sigma| + \log|K_{mm}|\right).$$

The Kullback-Leibler divergence regarding the Pólya-Gamma variables is also computed in closed form. Using $q(\omega_i) = \cosh\left(\frac{c_i}{2}\right)\exp\left(-\frac{c_i^2}{2}\omega_i\right)\mathrm{PG}(\omega_i|1,0)$ and $p(\omega_i) = \mathrm{PG}(\omega_i|1,0)$, we obtain

$$\begin{aligned}
&\mathrm{KL}(q(\boldsymbol{\omega}))||p(\boldsymbol{\omega})) = \mathbb{E}_{q(\boldsymbol{\omega})}\left[\log q(\boldsymbol{\omega}) - \log p(\boldsymbol{\omega})\right] \\
&= \sum_i \left(\mathbb{E}_{q(\omega_i)}\left[\log\left(\cosh\left(\frac{c_i}{2}\right)\exp\left(-\frac{c_i^2}{2}\omega_i\right)\mathrm{PG}(\omega_i|1,0)\right)\right] - \mathbb{E}_{q(\omega_i)}\left[\log\mathrm{PG}(\omega_i|1,0)\right]\right) \\
&= \sum_i \Big(\log\cosh\left(\frac{c_i}{2}\right) - \frac{c_i}{4}\tanh\left(\frac{c_i}{2}\right) + \mathbb{E}_{q(\omega_i)}\left[\log\mathrm{PG}(\omega_i|1,0)\right] \\
&\quad - \mathbb{E}_{q(\omega_i)}\left[\log\mathrm{PG}(\omega_i|1,0)\right]\Big) \\
&= \sum_i \left(\log\cosh\left(\frac{c_i}{2}\right) - \frac{c_i}{4}\tanh\left(\frac{c_i}{2}\right)\right).
\end{aligned}$$

Remarkably, all intractable expectations cancel out, which would not have been the case if we assumed a fully parameterized Pólya-Gamma distribution $\mathrm{PG}(\omega_i|b_i,c_i)$ as variational family. In Section 3.3, we found that we only have to consider the restricted variational class with $b_i = 1$.

Summing all terms results in the final lower bound

$$\begin{aligned}
\mathcal{L}(\boldsymbol{c},\boldsymbol{\mu},\Sigma) \stackrel{\mathsf{c}}{=} \frac{1}{2}\Bigg(&\log|\Sigma| - \log|K_{mm}|) - \mathrm{tr}(K_{mm}^{-1}\Sigma) - \boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} \\
&+ \sum_i \Big\{y_i\boldsymbol{\kappa}_i\boldsymbol{\mu} - \theta_i\widetilde{K}_{ii} - \theta_i\boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \theta_i\boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu} + c_i^2\theta_i \\
&- 2\log\cosh\left(\frac{c_i}{2}\right)\Big\}\Bigg).
\end{aligned}$$

A.2 Variational updates

Local parameters The derivative of the variational bound (3.6) w.r.t. the local parameter c_i is

$$\begin{aligned}
\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}c_i} &= \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}c_i}\left\{\theta_i\left(-\widetilde{K}_{ii} - \boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu} + c_i^2\right) - 2\log\cosh\frac{c_i}{2}\right\} \\
&= \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}c_i}\left\{\frac{1}{2c_i}\tanh\left(\frac{c_i}{2}\right)\left(-\widetilde{K}_{ii} - \boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu} + c_i^2\right) - 2\log\cosh\frac{c_i}{2}\right\} \\
&= \frac{\mathrm{d}}{\mathrm{d}c_i}\Bigg\{\frac{1}{4c_i}\tanh\left(\frac{c_i}{2}\right)\Bigg(\underbrace{-\widetilde{K}_{ii} - \boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top - \boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu}}_{=:-A_i}\Bigg) + \frac{c_i}{4}\tanh\left(\frac{c_i}{2}\right) \\
&\quad - \log\cosh\frac{c_i}{2}\Bigg\} \\
&= \left(\frac{A_i}{4c_i^2} - \frac{1}{4}\right)\tanh\left(\frac{c_i}{2}\right) - \frac{1}{2}\left(\frac{A_i}{4c_i} - \frac{c_i}{4}\right)\left(1 - \tanh^2(\frac{c_i}{2})\right) \\
&= U(c_i)\left(\frac{c_i}{2}\left(1 - \tanh^2(\frac{c_i}{2})\right) - \tanh\left(\frac{c_i}{2}\right)\right),
\end{aligned}$$

where $U(c_i) = \frac{\Sigma_{ii}+\mu_i^2}{4c_i^2} - \frac{1}{4}$.

The gradient equals zero in two case. First, in the case $U(c_i) = 0$ which leads to[1]

$$c_i = \sqrt{\widetilde{K}_{ii} + \boldsymbol{\kappa}_i\Sigma\boldsymbol{\kappa}_i^\top + \boldsymbol{\mu}^\top\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu}},$$

which is always valid since κ, Σ and $\widetilde{K}$ are definite positive matrices. The second case consists of the right hand side of the product being zero, which leads to $c_i = 0$. The second derivative reveals that the first case always corresponds to a maximum and the second case to a minimum. Hence, we only consider the first case.

Global parameters We first compute the Euclidean gradients of the variational bound (3.6) w.r.t. the global parameters $\boldsymbol{\mu}$ and Σ. We obtain

$$\begin{aligned}
\frac{d\mathcal{L}}{d\boldsymbol{\mu}} &= \frac{1}{2}\frac{d}{d\boldsymbol{\mu}}\left(-\boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} + \boldsymbol{y}^\top\kappa\boldsymbol{\mu} - \boldsymbol{\mu}^\top\kappa^\top\Theta\kappa\boldsymbol{\mu}\right) \\
&= \frac{1}{2}\left(-2K_{mm}^{-1}\boldsymbol{\mu} + \kappa^\top\boldsymbol{y} - 2\kappa^\top\Theta\kappa\boldsymbol{\mu}\right) \\
&= -\left(K_{mm}^{-1} + \kappa^\top\Theta\kappa\right)\boldsymbol{\mu} + \frac{1}{2}\kappa^\top\boldsymbol{y},
\end{aligned} \tag{A.1}$$

and

$$\begin{aligned}
\frac{d\mathcal{L}}{d\Sigma} &= \frac{1}{2}\frac{d}{d\Sigma}\left(\log|\,\Sigma\,| - \mathrm{tr}(K_{mm}^{-1}\Sigma) - \mathrm{tr}(\kappa^\top\Theta\kappa\Sigma)\right) \\
&= \frac{1}{2}\left(\Sigma^{-1} - K_{mm}^{-1} - \kappa^\top\Theta\kappa\right).
\end{aligned} \tag{A.2}$$

[1] We omit the negative solution since $\mathrm{PG}(b,c) = \mathrm{PG}(b,-c)$.

We now compute the natural gradients w.r.t. natural parameterization of the variational Gaussian distribution, i.e the parameters $\boldsymbol{\eta}_1 := \Sigma^{-1}\boldsymbol{\mu}$ and $\eta_2 = -\frac{1}{2}\Sigma^{-1}$. For a Gaussian distribution, properties of the Fisher information matrix expose the following way of computing the natural gradient. The natural gradient w.r.t. the natural parameters can be expressed in terms of the Euclidean gradient w.r.t. the mean and covariance parameters (Hensman, Fusi, and Lawrence, 2013). It holds that

$$\widetilde{\nabla}_{(\boldsymbol{\eta}_1,\eta_2)}\mathcal{L}(\eta) = \big(\nabla_\mu \mathcal{L}(\eta) - 2\nabla_\Sigma \mathcal{L}(\eta)\mu,\ \nabla_\Sigma \mathcal{L}(\eta)\big), \tag{A.3}$$

where $\widetilde{\nabla}$ denotes the natural gradient and ∇ the Euclidean gradient. Substituting the Euclidean gradients (A.2) and (A.1) in to equation (A.3) we obtain the natural gradients

$$\begin{aligned}\widetilde{\nabla}_{\eta_2}\mathcal{L} &= \frac{1}{2}\left(-2\eta_2 - K_{mm}^{-1} - \kappa^\top \Theta \kappa\right) \\ &= -\eta_2 - \frac{1}{2}\left(K_{mm}^{-1} + \kappa^\top \Theta \kappa\right)\end{aligned}$$

and

$$\begin{aligned}\widetilde{\nabla}_{\boldsymbol{\eta}_1}\mathcal{L} = &-\left(K_{mm}^{-1} + \kappa^\top \Theta \kappa\right)\left(-\frac{1}{2}\eta_2^{-1}\boldsymbol{\eta}_1\right) + \frac{1}{2}\kappa^\top \boldsymbol{y} \\ &- 2\left(-\eta_2 - \frac{1}{2}\left(K_{mm}^{-1} + \kappa^\top \Theta \kappa\right)\right)\left(-\frac{1}{2}\eta_2^{-1}\boldsymbol{\eta}_1\right) \\ = &\frac{1}{2}\kappa^\top \boldsymbol{y} - \boldsymbol{\eta}_1.\end{aligned}$$

A.3 Variational bound by Gibbs and MacKay

When using the full GP representation in our model and not the sparse approximation, the bound in our model is equal to the bound used by Gibbs and MacKay (2000). We provide a proof in the following.

Applying our variational inference approach to the joint distribution (3.4) gives the variational bound

$$\begin{aligned}\log p(\boldsymbol{y}|\,\boldsymbol{f}) &\geq \mathbb{E}_{q(\omega)}\left[\log p(\boldsymbol{y}|\,\boldsymbol{f},\boldsymbol{\omega})\right] - \mathrm{KL}(q(\boldsymbol{\omega})|p(\boldsymbol{\omega})) \\ &= \mathbb{E}_{q(\omega)}\left[\frac{1}{2}\boldsymbol{y}^\top \boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top \Omega \boldsymbol{f}\right] - n\log(2) - \mathrm{KL}(q(\boldsymbol{\omega})|p(\boldsymbol{\omega})) \\ &= \frac{1}{2}\boldsymbol{y}^\top \boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top \Theta \boldsymbol{f} - n\log(2) + \sum_{i=1}^{n}\left(\frac{c_i^2}{2}\theta_i - \log\cosh(c_i/2)\right).\end{aligned}$$

Gibbs and MacKay, 2000 employ the following inequality on the logistic function

$$\sigma(z) \geq \sigma(c)\exp\left(\frac{z-c}{2} - \frac{\sigma(c)-1/2}{2c}(z^2 - c^2)\right).$$

Using this bound in the setting of GP classification yields the following lower bound on the log-likelihood,

$$
\begin{aligned}
\log p(\boldsymbol{y}|\, \boldsymbol{f}) =& \sum_{i=1}^{n} \log \sigma(y_i f_i) \\
\geq& \sum_{i=1}^{n} \left(\log \sigma(c_i) + \frac{y_i f_i - c_i}{2} - \frac{\sigma(c_i) - 1/2}{2c_i}((y_i f_i)^2 - c_i^2) \right) \\
=& \sum_{i=1}^{n} \left(-\log \cosh(c_i/2) - \log(2) + \frac{y_i f_i}{2} - \frac{\sigma(c_i) - 1/2}{2c_i}(f_i^2 - c_i^2) \right) \\
=& \sum_{i=1}^{n} \left(-\log \cosh(c_i/2) - \log(2) + \frac{y_i f_i}{2} - \frac{1}{4c_i} \tanh(c_i/2)(f_i^2 - c_i^2) \right) \\
=& \sum_{i=1}^{n} \left(-\log \cosh(c_i/2) - \log(2) + \frac{y_i f_i}{2} - \frac{1}{2}\theta_i(f_i^2 - c_i^2) \right) \\
=& \frac{1}{2}\boldsymbol{y}^\top \boldsymbol{f} - \frac{1}{2}\boldsymbol{f}^\top \Theta \boldsymbol{f} - n \log(2) + \sum_{i=1}^{n} \left(\frac{c_i^2}{2}\theta_i - \log \cosh(c_i/2) \right),
\end{aligned}
$$

where we made use of the fact that $\sigma(x) - 1/2 = \tanh(x/2)/2$. This concludes the proof.

A.4 Additional performance plots

On the next pages, we show all time vs. prediction performance plots for the datasets presented in Table 3.1, which are not already shown in Chapter 3.

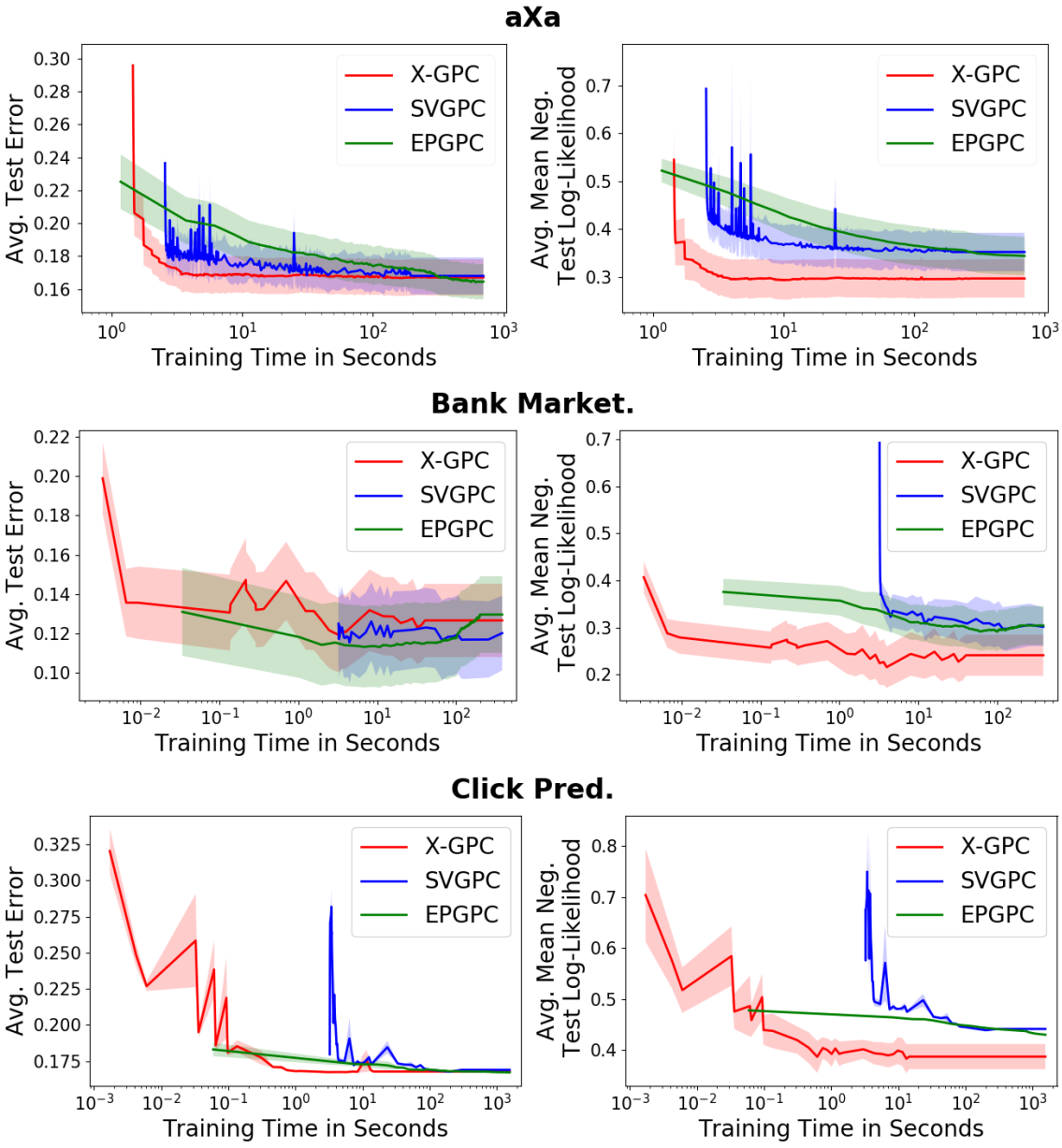

Figure A.1: Average negative test log-likelihood and average test prediction error as a function of training time measured in seconds (on a $\log_{10}$ scale).

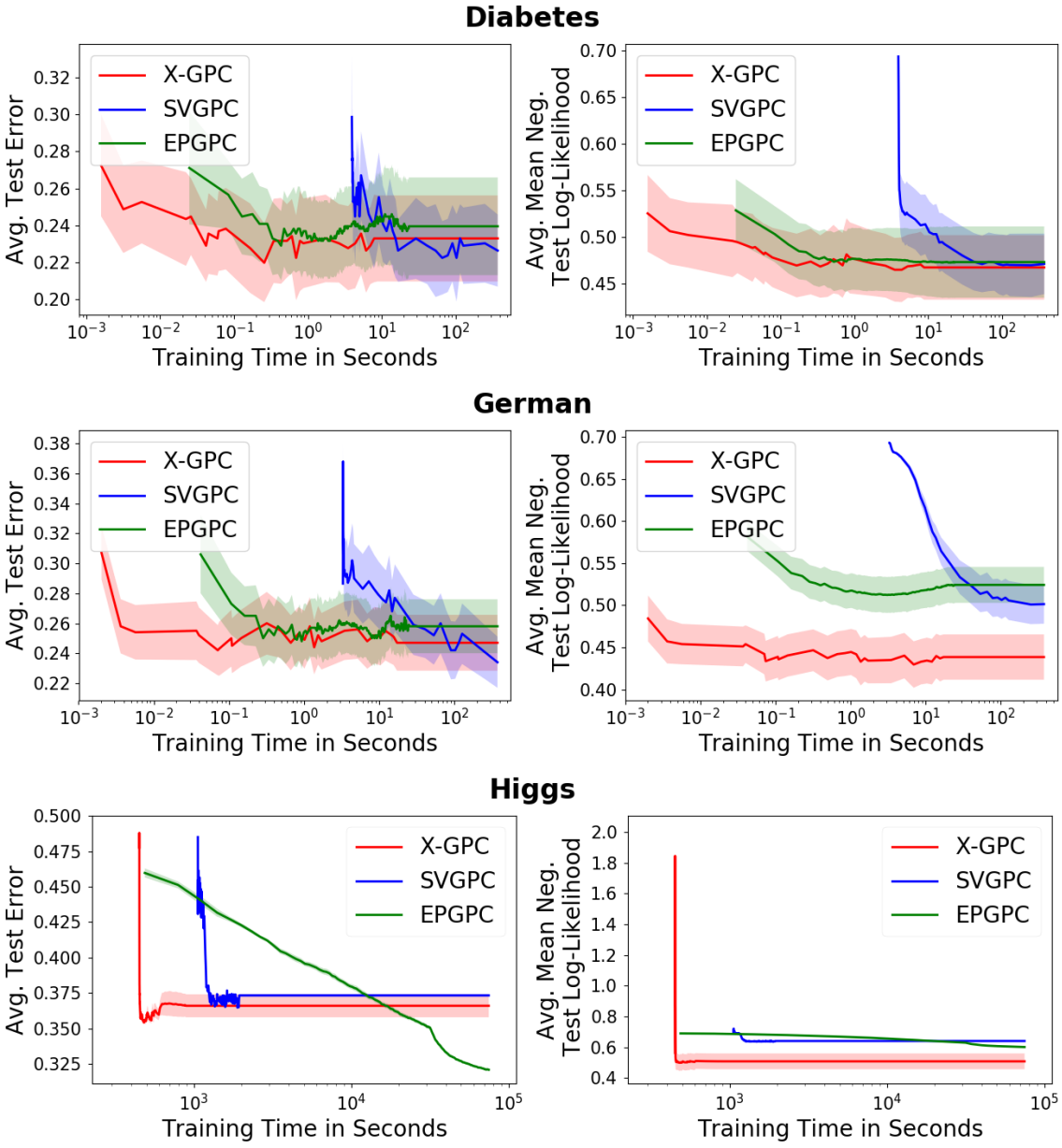

Figure A.2: Average negative test log-likelihood and average test prediction error as a function of training time measured in seconds (on a $\log_{10}$ scale). For the dataset Higgs, EPGPC exceeded the time budget of 10^5 seconds ($\approx$ 28 h).

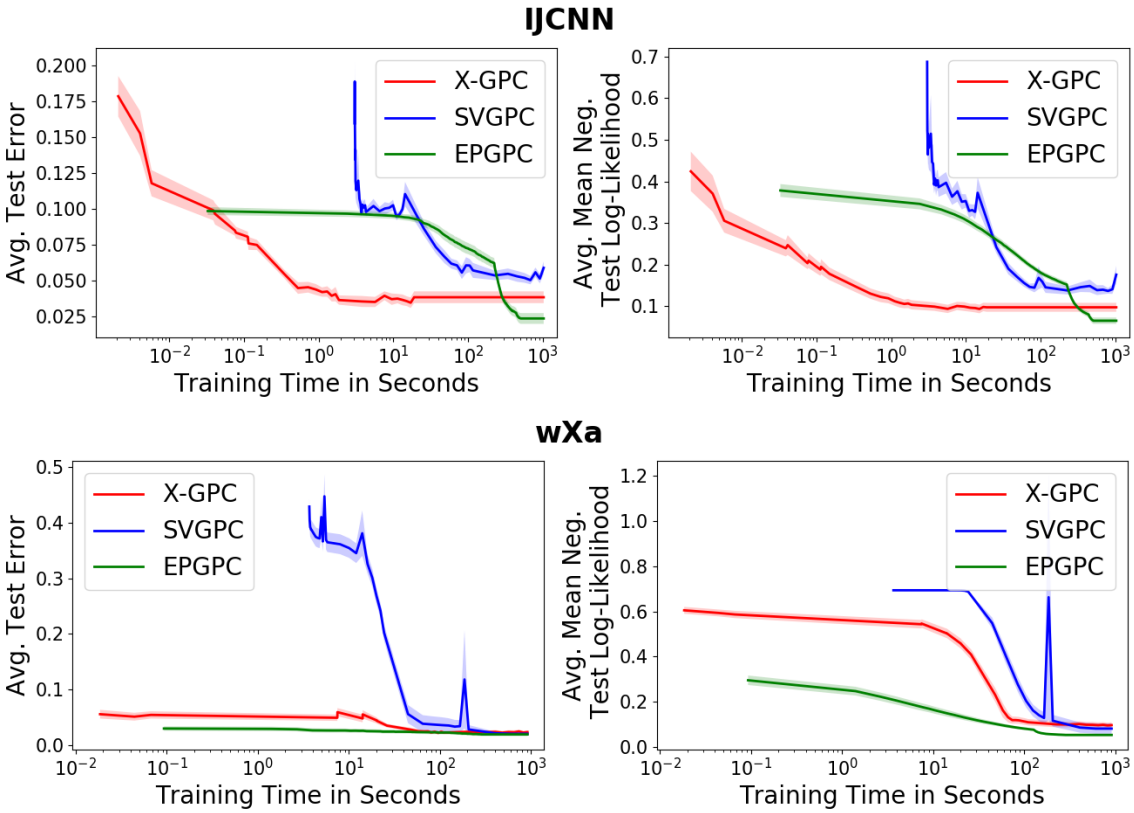

Figure A.3: Average negative test log-likelihood and average test prediction error as a function of training time measured in seconds (on a $\log_{10}$ scale).

Appendix B

Multi-class Gaussian process classification

B.1 Reparametrization of the Pólya-Gamma variables

For this augmentation step, we start with the likelihood from step 2, Eq. 4.7. We apply the augmentation of the sigmoid (4.8) and obtain the Pólya-Gamma augmented likelihood

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i, \boldsymbol{n}_i, \tilde{\omega}_i, \boldsymbol{\omega}_i) = \frac{1}{2}\exp\left(\frac{f_i^k}{2} - \frac{(f_i^k)^2}{2}\tilde{\omega}_i\right)$$
$$\times \prod_{c=1}^{C} 2^{-n_i^c}\exp\left(-\frac{n_i^c f_i^c}{2} - \frac{(f_i^c)^2}{2}\omega_i^c\right), \tag{B.1}$$

where we impose the prior distributions

$$p(\tilde{\omega}_i) = \text{PG}(1, 0)$$
$$p(\boldsymbol{\omega}_i|\boldsymbol{n}_i) = \prod_c^C \text{PG}(\omega_i^c|n_i^c, 0).$$

We simplify this expression by merging the Pólya-Gamma variables ω_i^k and $\tilde{\omega}_i$. To this end, we use a one hot-encoding of $\boldsymbol{y} \in \{0, \ldots, C\}^N$ and define $y' \in \{0, 1\}^{C\times N}$ by

$$y'^c_i = \begin{cases} 1 \text{ for } y_i = c \\ 0 \text{ otherwise.} \end{cases}.$$

Furthermore, we use the fact that the sum of two Pólya-Gamma variables $\omega_3 = \omega_1 + \omega_2$ is again Pólya-Gamma distributed, given by $\omega_3 \sim \text{PG}(b_1+b_2, c)$, where $\omega_1 \sim \text{PG}(b_1, c)$, $\omega_2 \sim \text{PG}(b_2, c)$. This leads to a simplification of the augmented likelihood (B.1) which

is given by

$$p(y_i = k|\boldsymbol{f}_i, \lambda_i, \boldsymbol{n}_i, \boldsymbol{\omega}_i) = \\ \prod_{c=1}^{C} 2^{-(y'^c_i + n^c_i)} \exp\left(\frac{(y'^c_i - n^c_i) f^c_i}{2} - \frac{(f^c_i)^2}{2}\omega^c_i\right),$$

with the (merged) prior distributions

$$p(\boldsymbol{\omega}_i|\boldsymbol{n}_i, y_i) = \prod_{c=1}^{C} \mathrm{PG}(\omega^c_i|y'^c_i + n^c_i, 0).$$

B.2 Subsampling the classes (extreme classification version)

The extreme classification version of our algorithm is described in Section 4.3.2 and is summarized in Alg. 4.

Algorithm 4 Conjugate multi-class Gaussian process classification with class subsampling

Input: data $\boldsymbol{X}$,$\boldsymbol{y}$, minibatch size $|\mathcal{S}|$and $|\mathcal{B}|$
Output: variational posterior GPs $p(u^c|\mu^c, \Sigma^c)$
Set the learning rate schedules ρ_t, ρ^h_t appropriately
Initialize all variational parameters and hyperparameters
Select M inducing points locations (e.g. kMeans)
for iteration $t = 1, 2, \ldots$ **do**
    `# Sample minibatch:`
    Sample a minibatch of the data $\mathcal{S} \subset \{1, \ldots, N\}$
    Sample a set of labels $\mathcal{K} \subset \{1, \ldots, C\}$
    `# Local variational updates`
    for $i \in \mathcal{S}$ **do**
        Update $(\alpha_i, \gamma^c_i)_{c\in\mathcal{K}}$ (Eq. 4.11,4.17)
        for $c \in \mathcal{K}$ **do**
            Update b^c_i (Eq. 4.13)
    `# Global variational GP updates`
    for $c \in \mathcal{K}$ **do**
        $\mu^c \leftarrow (1 - \rho_t)\mu^c + \rho_t\hat{\mu}^c$ (Eq. 4.15)
        $\Sigma^c \leftarrow (1 - \rho_t)\Sigma^c + \rho_t\hat{\Sigma}^c$ (Eq. 4.16)
    `# Hyperparameter updates`
    Gradient step $h \leftarrow h + \rho^h_t \nabla_h \mathcal{L}$

Appendix C

Bayesian support vector machine

C.1 Derivation of the variational objective

In the following, we provide the details of the derivation of the variational objective (ELBO) used in Section 5.3. We first consider the term

$$
\begin{aligned}
\mathcal{L}_1 &:= \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{u})}\left[\log p(\boldsymbol{y}, \lambda|\boldsymbol{f})\right] \\
&= \sum_{i=1}^{n} \mathbb{E}_{p(f_i|\boldsymbol{u})}\left[\log p(y_i, \lambda_i|f_i)\right] \\
&= \sum_{i=1}^{n} \mathbb{E}_{p(f_i|\boldsymbol{u})}\left[\log\left((2\pi\lambda_i)^{-\frac{1}{2}}\exp\left(-\frac{1}{2}\frac{(1+\lambda_i - y_i f_i)^2}{\lambda_i}\right)\right)\right] \\
&\overset{\mathsf{C}}{=} -\frac{1}{2}\sum_{i=1}^{n} \mathbb{E}_{p(f_i|\boldsymbol{u})}\left[\log\lambda_i + \frac{(1+\lambda_i - y_i f_i)^2}{\lambda_i}\right] \\
&= -\frac{1}{2}\sum_{i=1}^{n}\left(\log\lambda_i + \frac{1}{\lambda_i}\mathbb{E}_{p(f_i|\boldsymbol{u})}\left[(1+\lambda_i - y_i f_i)^2\right]\right) \\
&= -\frac{1}{2}\sum_{i=1}^{n}\left(\log\lambda_i + \frac{1}{\lambda_i}\left(\widetilde{K}_{ii} + \left(1+\lambda_i - y_i \mathbf{K}_{im}K_{mm}^{-1}\boldsymbol{u}\right)^2\right)\right)
\end{aligned}
$$

The ELBO is

$$
\begin{aligned}
\mathcal{L} &= \mathbb{E}_q\left[\mathcal{L}_1\right] + \mathbb{E}_q\left[\log p(\boldsymbol{u})\right] - \mathbb{E}_q\left[\log q(\boldsymbol{\lambda}, \boldsymbol{u})\right] \\
&= -\frac{1}{2}\sum_{i=1}^{n}\mathbb{E}_q\left[\log\lambda_i + \frac{1}{\lambda_i}\left(\widetilde{K}_{ii} + \left(1+\lambda_i - y_i\mathbf{K}_{im}K_{mm}^{-1}\boldsymbol{u}\right)^2\right)\right] \\
&\quad - \mathrm{KL}\left(q(\boldsymbol{u})||p(\boldsymbol{u})\right) - \mathbb{E}_{q(\boldsymbol{\lambda})}\left[\log q(\boldsymbol{\lambda})\right].
\end{aligned}
$$

Using the abbreviation $\boldsymbol{\kappa}_i = \mathbf{K}_{im} K_{mm}^{-1}$, the first expectation term simplifies to

$$
\begin{aligned}
&\mathbb{E}_q \left[\log \lambda_i + \frac{1}{\lambda_i} \left(\widetilde{K}_{ii} + \left(1 + \lambda_i - y_i \mathbf{K}_{im} K_{mm}^{-1} \boldsymbol{u} \right)^2 \right) \right] \\
&= \mathbb{E}_q[\log \lambda_i] + \mathbb{E}_q \left[\lambda_i^{-1} \left(\widetilde{K}_{ii} + 1 + \lambda_i^2 + \underbrace{y_i^2}_{=1} (\boldsymbol{\kappa}_i \boldsymbol{u})^2 + 2\lambda_i - 2 y_i \boldsymbol{\kappa}_i \boldsymbol{u} - 2\lambda_i y_i \boldsymbol{\kappa}_i \boldsymbol{u} \right) \right] \\
&\overset{\mathsf{C}}{=} \mathbb{E}_{q(\lambda_i)}[\log \lambda_i] + \frac{1}{\sqrt{\alpha_i}} \left(\widetilde{K}_{ii} + 1 + \lambda_i^2 + (\boldsymbol{\kappa}_i \boldsymbol{\mu})^2 + \boldsymbol{\kappa}_i \zeta \boldsymbol{\kappa}_i^\top - 2 y_i \boldsymbol{\kappa}_i \boldsymbol{\mu} \right) + \mathbb{E}_{q(\lambda_i)}[\lambda_i] \\
&\quad - 2 y_i \boldsymbol{\kappa}_i \boldsymbol{\mu} \\
&= \mathbb{E}_{q(\lambda_i)}[\log \lambda_i] + \frac{1}{\sqrt{\alpha_i}} \left(\widetilde{K}_{ii} + (1 - y_i \boldsymbol{\kappa}_i \boldsymbol{\mu})^2 + \lambda_i^2 + \boldsymbol{\kappa}_i \zeta \boldsymbol{\kappa}_i^\top \right) + \mathbb{E}_{q(\lambda_i)}[\lambda_i] - 2 y_i \boldsymbol{\kappa}_i \boldsymbol{\mu}.
\end{aligned}
$$

The entropy of $q(\lambda_i)$ is

$$
\begin{aligned}
\mathbb{E}_{q(\lambda_i)} \left[\log q(\lambda_i) \right] &\overset{\mathsf{C}}{=} \mathbb{E}_{q(\lambda_i)} \left[-\frac{1}{4} \log(\alpha_i) - \frac{1}{2} \log(\lambda_i) - \log(\mathrm{B}_{\frac{1}{2}}(\sqrt{\alpha_i})) - \frac{1}{2} \left(\lambda_i + \frac{\alpha_i}{\lambda_i} \right) \right] \\
&\overset{\mathsf{C}}{=} -\frac{1}{4} \log(\alpha_i) - \frac{1}{2} \mathbb{E}_{\alpha_i} \left[\log(\lambda_i) \right] - \log(\mathrm{B}_{\frac{1}{2}}(\sqrt{\alpha_i})) - \frac{1}{2} \mathbb{E}_{\alpha_i} \left[\lambda_i \right] \\
&\quad - \frac{\alpha_i}{2} \mathbb{E}_{\alpha_i} \left[\frac{1}{\lambda_i} \right] \\
&\overset{\mathsf{C}}{=} -\frac{1}{4} \log(\alpha_i) - \frac{1}{2} \mathbb{E}_{\alpha_i} \left[\log(\lambda_i) \right] - \log(\mathrm{B}_{\frac{1}{2}}(\sqrt{\alpha_i})) - \frac{1}{2} \mathbb{E}_{\alpha_i} \left[\lambda_i \right] - \frac{\sqrt{\alpha_i}}{2},
\end{aligned}
$$

where $\mathrm{B}_{\frac{1}{2}}(.)$ is the modified Bessel function with parameter $\frac{1}{2}$ (Jørgensen, 2012). By summing the terms the remaining expectations cancel out and we obtain

$$
\begin{aligned}
\mathcal{L} &\overset{\mathsf{c}}{=} \sum_{i=1}^{n} \Big\{ -\frac{1}{2}\mathbb{E}_{q(\lambda_i)}[\log \lambda_i] - \frac{1}{2\sqrt{\alpha_i}} \left(\widetilde{K}_{ii} + (1 - y_i \boldsymbol{\kappa}_i \boldsymbol{\mu})^2 + \lambda_i^2 + \boldsymbol{\kappa}_i \zeta \boldsymbol{\kappa}_i^\top \right) - \frac{1}{2}\mathbb{E}_{q(\lambda_i)}[\lambda_i] \\
&\quad + y_i \boldsymbol{\kappa}_i \boldsymbol{\mu} + \frac{1}{4}\log(\alpha_i) + \frac{1}{2}\mathbb{E}_{q(\lambda_i)}\left[\log(\lambda_i)\right] + \log(\mathrm{B}_{\frac{1}{2}}(\sqrt{\alpha_i})) \\
&\quad + \frac{1}{2}\mathbb{E}_{q(\lambda_i)}\left[\lambda_i\right] + \frac{\sqrt{\alpha_i}}{2} \Big\} - \mathrm{KL}\left(q(\boldsymbol{u}) || p(\boldsymbol{u})\right) \\
&= \sum_{i=1}^{n} \Big\{ -\frac{1}{2\sqrt{\alpha_i}} \left(\widetilde{K}_{ii} + (1 - y_i \boldsymbol{\kappa}_i \boldsymbol{\mu})^2 + \lambda_i^2 + \boldsymbol{\kappa}_i \zeta \boldsymbol{\kappa}_i^\top - \alpha_i \right) + y_i \boldsymbol{\kappa}_i \boldsymbol{\mu} \\
&\quad + \frac{1}{4}\log(\alpha_i) + \log(\mathrm{B}_{\frac{1}{2}}(\sqrt{\alpha_i})) \Big\} - \mathrm{KL}\left(q(\boldsymbol{u}) || p(\boldsymbol{u})\right) \\
&\overset{\mathsf{c}}{=} \sum_{i=1}^{n} \Big\{ -\frac{1}{2\sqrt{\alpha_i}} \left(\widetilde{K}_{ii} + (1 - y_i \boldsymbol{\kappa}_i \boldsymbol{\mu})^2 + \lambda_i^2 + \boldsymbol{\kappa}_i \zeta \boldsymbol{\kappa}_i^\top - \alpha_i \right) + y_i \boldsymbol{\kappa}_i \boldsymbol{\mu} \\
&\quad + \frac{1}{4}\log(\alpha_i) + \log(\mathrm{B}_{\frac{1}{2}}(\sqrt{\alpha_i})) \Big\} + \frac{1}{2}\log|\zeta| - \frac{1}{2}\mathrm{tr}(K_{mm}^{-1}\zeta) - \frac{1}{2}\boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} \\
&= \frac{1}{2}\log|\zeta| - \frac{1}{2}\mathrm{tr}(K_{mm}^{-1}\zeta) - \frac{1}{2}\boldsymbol{\mu}^\top K_{mm}^{-1}\boldsymbol{\mu} + y^\top \kappa \boldsymbol{\mu} + \sum_{i=1}^{n} \Big\{ \log(\mathrm{B}_{\frac{1}{4}}(\sqrt{\alpha_i})) \\
&\quad + \frac{1}{2}\log(\alpha_i) - \frac{1}{2}\alpha_i^{-\frac{1}{2}} \left(1 - \alpha_i - 2 y_i \kappa_{i\cdot} \boldsymbol{\mu} + \left(\kappa(\boldsymbol{\mu}\boldsymbol{\mu}^\top + \zeta)\kappa^\top + \widetilde{K} \right)_{ii} \right) \Big\}.
\end{aligned}
$$

C.2 Euclidean and natural gradients of the variational objective

First, we compute the standard Euclidean gradients of $\mathcal{L}$. The derivative w.r.t. the mean and covariance matrix are

$$
\begin{aligned}
\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\zeta} &= \frac{1}{2}\left(\frac{\mathrm{d}}{\mathrm{d}\zeta}\log|\zeta| - \frac{\mathrm{d}}{\mathrm{d}\zeta}\mathrm{tr}(K_{mm}^{-1}\zeta) \right) + \sum_{i=1}^{N} -\frac{1}{2\sqrt{\alpha_i}} \frac{\mathrm{d}}{\mathrm{d}\zeta} y_i \boldsymbol{\kappa}_i \zeta \boldsymbol{\kappa}_i^\top y_i \\
&= \frac{1}{2}\left(\zeta^{-1}\right)^T - \frac{1}{2}\left(K_{mm}^{-1}\right)^T - \frac{1}{2} Y^2 \kappa^\top A^{-\frac{1}{2}} \kappa \\
&= \frac{1}{2}\left(\zeta^{-1} - K_{mm}^{-1} - \kappa^\top A^{-\frac{1}{2}} \kappa \right) \\
&=: \mathcal{L}'_\zeta,
\end{aligned}
$$

with $A = \mathrm{diag}(\boldsymbol{\alpha})$ and

$$\begin{aligned}
\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\boldsymbol{\mu}} &= -\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\mu}}\boldsymbol{\mu}^T K_{mm}^{-1}\boldsymbol{\mu} + \sum_{i=1}^{N}\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\mu}} y_i\boldsymbol{\kappa}_i\boldsymbol{\mu} + \frac{1}{2\sqrt{\alpha_i}}\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\mu}}(1 - y_i\boldsymbol{\kappa}_i\boldsymbol{\mu})^2 \\
&= -K_{mm}^{-1}\boldsymbol{\mu} + \sum_{i=1}^{N} y_i\boldsymbol{\kappa}_i + \frac{1}{\sqrt{\alpha_i}}\left(y_i\boldsymbol{\kappa}_i^\top - y_i^2\boldsymbol{\kappa}_i^\top\boldsymbol{\kappa}_i\boldsymbol{\mu}\right) \\
&= -K_{mm}^{-1}\boldsymbol{\mu} + \kappa^\top y + \kappa^\top Y\alpha^{-\frac{1}{2}} + \kappa^\top A^{-\frac{1}{2}}\kappa\boldsymbol{\mu} \\
&= -\left(K_{mm}^{-1} + \kappa^\top A^{-\frac{1}{2}}\kappa\right)\boldsymbol{\mu} + \kappa^\top Y(\boldsymbol{\alpha}^{-\frac{1}{2}} + 1) \\
&=: \mathcal{L}_{\boldsymbol{\mu}\prime}.
\end{aligned}$$

The derivative w.r.t. parameter $\boldsymbol{\alpha}_i$ of the generalized inverse Gaussian distribution is

$$\begin{aligned}
\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\alpha_i} &= \frac{1}{4}\frac{\mathrm{d}}{\mathrm{d}\alpha_i}\log(\alpha_i) + \frac{\mathrm{d}}{\mathrm{d}\alpha_i}\log(K_{\frac{1}{2}}(\sqrt{\alpha_i})) + \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}\alpha_i}\sqrt{\alpha_i} \\
&\quad - \frac{(1 - y_i\boldsymbol{\kappa}_i\boldsymbol{\mu})^2 + y_i(\boldsymbol{\kappa}_i\zeta\boldsymbol{\kappa}_i^\top + \widetilde{K}_{ii})y_i}{2}\frac{\mathrm{d}}{\mathrm{d}\alpha_i}\frac{1}{\sqrt{\alpha_i}} \\
&= \frac{1}{4\alpha_i} - (\frac{1}{4\alpha_i} + \frac{1}{2\sqrt{\alpha_i}}) + \frac{1}{4\sqrt{\alpha_i}} + \frac{(1 - y_i\boldsymbol{\kappa}_i\boldsymbol{\mu})^2 + y_i(\boldsymbol{\kappa}_i\zeta\boldsymbol{\kappa}_i^\top + \widetilde{K}_{ii})y_i}{4\sqrt{\alpha_i}^3} \\
&= \frac{(1 - y_i\boldsymbol{\kappa}_i\boldsymbol{\mu})^2 + y_i(\boldsymbol{\kappa}_i\zeta\boldsymbol{\kappa}_i^\top + \widetilde{K}_{ii})y_i}{4\sqrt{\alpha_i}^3} - \frac{1}{4\sqrt{\alpha_i}}.
\end{aligned}$$

The natural gradient is computed by pre-multiplying the Euclidean gradient with the inverse Fisher information matrix (Amari and Nagaoka, 2007). Applied to a Gaussian distribution this leads to the following expressions for the natural gradient w.r.t. the natural parameters (Amari and Nagaoka, 2007),

$$\widetilde{\nabla}_{(\eta_1,\eta_2)}\mathcal{L}(\eta) = \left(\mathcal{L}_{\boldsymbol{\mu}\prime}(\eta) - 2\mathcal{L}'_\zeta(\eta)\boldsymbol{\mu},\ \mathcal{L}'_\zeta(\eta)\right).$$

Using the identities $\eta_1 = \zeta^{-1}\boldsymbol{\mu}$ and $\eta_2 = -\frac{1}{2}\zeta^{-1}$, we obtain

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{\mu}\prime}(\eta) &= \frac{1}{2}\left(K_{mm}^{-1} + \kappa^\top A^{-\frac{1}{2}}\kappa\right)\eta_2^{-1}\eta_1 + \kappa^\top Y(\boldsymbol{\alpha}^{-\frac{1}{2}} + 1) \\
\mathcal{L}'_\zeta(\eta) &= \frac{1}{2}\left(-2\eta_2 - K_{mm}^{-1} - \kappa^\top A^{-\frac{1}{2}}\kappa\right) = -\frac{1}{2}(K_{mm}^{-1} + \kappa^\top A^{-\frac{1}{2}}\kappa) - \eta_2
\end{aligned}$$

Finally, this leads to the natural gradients with respect to the natural parameters

$$\begin{aligned}
\widetilde{\nabla}_{\eta_1}\mathcal{L} &= \mathcal{L}_{\boldsymbol{\mu}\prime} - 2\mathcal{L}'_\zeta\boldsymbol{\mu} \\
&= \frac{1}{2}\left(K_{mm}^{-1} + \kappa^\top A^{-\frac{1}{2}}\kappa\right)\eta_2^{-1}\eta_1 + \kappa^\top Y(\boldsymbol{\alpha}^{-\frac{1}{2}} + 1) \\
&\quad + \frac{1}{2}\left(-2\eta_2 - K_{mm}^{-1} - \kappa^\top A^{-\frac{1}{2}}\kappa\right)\eta_2^{-1}\eta_1 \\
&= \kappa^\top Y(\boldsymbol{\alpha}^{-\frac{1}{2}} + 1) - \eta_1,
\end{aligned}$$

and

$$\widetilde{\nabla}_{\eta_2}\mathcal{L} = \mathcal{L}'_\zeta = -\frac{1}{2}(K_{mm}^{-1} + \kappa^\top A^{-\frac{1}{2}}\kappa) - \eta_2.$$

C.3 Optimization of the kernel hyperparameters

We consider a general multiple kernel approach. Let $k(x, x') = \sum_j \gamma_j k_j(x, x', \theta_j)$ be the kernel function where θ_j denote the hyperparameters of the kernel function k_j (e.g. the length scale parameter of an squared exponential kernel) and γ_j the corresponding kernel weight. Let $\omega = \{\theta_j, \gamma_j\}_{j=1,\dots,J}$ be the collection of all hyperparameters. The derivative of the variational objective $\mathcal{L}$ w.r.t. to the hyperparameters is

$$\begin{aligned}\frac{\mathrm{d}L}{\mathrm{d}\omega} = -\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}\omega}\Big(&\log|K_{mm}| + \mathrm{tr}(K_{mm}^{-1}\zeta) + \boldsymbol{\mu}^T K_{mm}^{-1}\boldsymbol{\mu} - 2(1 + \boldsymbol{\alpha}^{-\frac{1}{2}^\top})Y\kappa\boldsymbol{\mu} \\ &+ \boldsymbol{\alpha}^{-\frac{1}{2}^\top}\mathrm{diag}\left(\kappa(\boldsymbol{\mu}\boldsymbol{\mu}^\top + \zeta)\kappa^\top + \widetilde{K}\right)\Big)\end{aligned}$$

Using the abbreviations $J_{**}^\omega = \frac{\mathrm{d}K_{**}}{\mathrm{d}\omega}$ and $\iota^\omega = \frac{\mathrm{d}\kappa}{\mathrm{d}\omega} = \left(J_{nm}^\omega - \kappa J_{mm}^\omega\right) K_{mm}^{-1}$ we obtain

$$\begin{aligned}\frac{\mathrm{d}L}{\mathrm{d}\omega} = -\frac{1}{2}\Big(&\mathrm{tr}\left(K_{mm}^{-1}J_{mm}^\omega\left(\mathbb{I} - K_{mm}^{-1}\zeta\right)\right) - \left(\boldsymbol{\mu}^\top K_{mm}^{-1}J_{mm}^\omega K_{mm}^{-1} + 2(1 + \boldsymbol{\alpha}^{-\frac{1}{2}^\top})Y\iota^\omega\right)\boldsymbol{\mu} \\ &+ \boldsymbol{\alpha}^{-\frac{1}{2}^\top}\mathrm{diag}\left[\kappa\left(\left(\boldsymbol{\mu}\boldsymbol{\mu}^\top + \zeta\right)\iota^{\omega\top} - J_{mn}^\omega\right) + \iota^\omega\left(\left(\boldsymbol{\mu}\boldsymbol{\mu}^\top + \zeta\right)\kappa^\top - K_{mn}\right) + J_{nn}^\omega\right]\Big).\end{aligned}$$

To compute the gradient w.r.t. to specific hyperparameters we only have to plug in the derivatives of the kernel function $\frac{\mathrm{d}K_{**}}{\mathrm{d}\omega}$ into the above formula.

Appendix D

Sparse Gaussian process linear mixed model

D.1 Convexity of the Objective Function

We prove that the objective function Eq. 6.4 is convex. Since the ℓ_1-norm regularizer is convex it is sufficient to show that $\ell(\boldsymbol{\beta}) = -\log \int_{\mathbb{R}^n_+} \mathcal{N}(\boldsymbol{f}; X\boldsymbol{\beta}, K)\,\mathrm{d}\boldsymbol{f}$ is convex in $\boldsymbol{\beta}$. Recall that a function g is log-convex, if g is strictly positive and $\log g$ is convex; log-concavity is defined analogously. In the following, we make use of a theorem that connects log-concave functions to their partial integrals over convex sets (Prékopa, 1973). Namely, for a log-concave function $g : \mathbb{R}^{n+m} \to \mathbb{R}$ and a convex subset $A \subset \mathbb{R}^n$, the function $h(\boldsymbol{x}) = \int_A g(\boldsymbol{x}, \boldsymbol{y})\mathrm{d}\boldsymbol{y}$ is log-concave in the entire space $\mathbb{R}^n$. Since $X\boldsymbol{\beta}$ is linear, it is sufficient to show that

$$g(\boldsymbol{\mu}) = -\log \int_{\mathbb{R}^n_+} \mathcal{N}(f; \boldsymbol{\mu}, K)\,\mathrm{d}f$$

is convex in $\boldsymbol{\mu}$. The multivariate Gaussian density $\mathcal{N}$ is log-concave in $(\boldsymbol{f}, \boldsymbol{\mu}) \in \mathbb{R}^{2n}$, since $\mathcal{N}(\boldsymbol{f}; \boldsymbol{\mu}, K) > 0$ for all $\boldsymbol{\mu}, \boldsymbol{f} \in \mathbb{R}^n$ and $\log \mathcal{N}$ is concave in $(\boldsymbol{f}, \boldsymbol{\mu})$. Therefore, $\int_{\mathbb{R}^n_+} \mathcal{N}(\boldsymbol{f}; \boldsymbol{\mu}, K)\,\mathrm{d}\boldsymbol{f}$ is log-concave in $\boldsymbol{\mu}$. The logarithm of a log-concave function is concave by definition. Thus, g is convex in $\boldsymbol{\mu}$ and therefore, Eq. 6.4 is convex in $\boldsymbol{\beta}$. QED.

D.2 Gradient and Hessian

In this section, we calculate the gradient and the Hessian of the log likelihood

$$\ell(\boldsymbol{\beta}) = \log \int_{\mathbb{R}^n_+} \mathcal{N}(\boldsymbol{f}|X\boldsymbol{\beta}, K)\mathrm{d}\boldsymbol{f}.$$

We use the notation from Chapter 6 and write $\boldsymbol{\mu}_p = \mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{\mu},K)}[\boldsymbol{f}]$, which is the mean of the truncated Gaussian (6.7) and $\boldsymbol{\mu}(\boldsymbol{\beta}) = X\boldsymbol{\beta}$. The gradient is given by

$$\nabla_\beta \ell(\boldsymbol{\beta}) = -\frac{\int_{\mathbb{R}_+^n} (\boldsymbol{f}-\boldsymbol{\mu})^\top K^{-1} \mathcal{N}(\boldsymbol{f};\boldsymbol{\mu},K)\mathrm{d}\boldsymbol{f}}{\int_{\mathbb{R}_+^n} \mathcal{N}(\boldsymbol{f};\boldsymbol{\mu},K)\mathrm{d}\boldsymbol{f}} X = -(\boldsymbol{\mu}_p - \boldsymbol{\mu})^\top K^{-1} X.$$

We now compute the Hessian. We first consider the Hessian matrix of $\ell(\boldsymbol{\mu})$, $B_{ij}(\boldsymbol{\mu}) = -\partial_{\boldsymbol{\mu}_i}\partial_{\boldsymbol{\mu}_j}\ell(\boldsymbol{\mu})$. The chain rule relates this object to the Hessian of $\ell(\boldsymbol{\beta})$, namely

$$H(\boldsymbol{\beta}) = XB(\boldsymbol{\mu})X$$

The problem therefore reduces to calculating $B(\boldsymbol{\mu})$, which is a $n \times n$-matrix, whereas the original Hessian $H(\boldsymbol{\beta})$ is $d \times d$.

To calculate $B(\boldsymbol{\mu})$, we define $J(\boldsymbol{\mu}) = \int_{\mathbb{R}_n^+} \exp\{-\frac{1}{2}(\boldsymbol{f}-\boldsymbol{\mu})^\top K^{-1}(\boldsymbol{f}-\boldsymbol{\mu})\}\mathrm{d}\boldsymbol{f}$. The Hessian is given by

$$B_{ij}(\boldsymbol{\mu}) = \frac{\partial_{\boldsymbol{\mu}_i}\partial_{\boldsymbol{\mu}_j} J(\boldsymbol{\mu})}{J(\boldsymbol{\mu})} - \frac{\partial_{\boldsymbol{\mu}_i} J(\boldsymbol{\mu})}{J(\boldsymbol{\mu})}\frac{\partial_{\boldsymbol{\mu}_j} J(\boldsymbol{\mu})}{J(\boldsymbol{\mu})}.$$

Note that this involves also the first derivatives of $J(\boldsymbol{\mu})$, that we have already calculated for the gradient. To proceed, we still need to calculate $\partial_{\boldsymbol{\mu}_i}\partial_{\boldsymbol{\mu}_j} J(\boldsymbol{\mu})$. To simplify the calculation, we introduce $\tilde{\boldsymbol{\mu}} = \boldsymbol{f} - \boldsymbol{\mu}$. As a consequence, $\partial_{\tilde{\boldsymbol{\mu}}_i} = -\partial_{\boldsymbol{\mu}_i}$. Furthermore,

$$\partial_{\boldsymbol{\mu}_i}\partial_{\boldsymbol{\mu}_j} \exp\{-\frac{1}{2}(\boldsymbol{f}-\boldsymbol{\mu})^\top K^{-1}(\boldsymbol{f}-\boldsymbol{\mu})\} = \left[K^{-1}\tilde{\boldsymbol{\mu}}\tilde{\boldsymbol{\mu}}^\top K^{-1} - K^{-1}\right]_{ij} \exp\{-\frac{1}{2}\tilde{\boldsymbol{\mu}}^\top K^{-1}\tilde{\boldsymbol{\mu}}\}.$$

Based on this identity, we derive $\frac{\partial_{\mu_i}\partial_{\mu_j} J(\boldsymbol{\mu})}{J(\boldsymbol{\mu})} = \left(K^{-1}K_p K^{-1} - K^{-1}\right)_{ij}$. For the remaining terms, we use our known result for the gradient, namely

$$\frac{\partial_{\boldsymbol{\mu}} J(\boldsymbol{\mu})}{J(\boldsymbol{\mu})} = \left(\mathbb{E}_{p(\boldsymbol{f}|\boldsymbol{\mu})}[(\boldsymbol{\mu}_p - \boldsymbol{\mu})^\top K^{-1}]\right) = (\boldsymbol{\mu}_p - \boldsymbol{\mu})^\top K^{-1}.$$

As a consequence,

$$\frac{\partial_{\boldsymbol{\mu}_i} J(\boldsymbol{\mu})}{J(\boldsymbol{\mu})}\frac{\partial_{\boldsymbol{\mu}_j} J(\boldsymbol{\mu})}{J(\boldsymbol{\mu})} = \left(K^{-1}\Delta\boldsymbol{\mu}\,\Delta\boldsymbol{\mu}^\top K^{-1}\right)_{ij}.$$

Above we defined $\Delta\boldsymbol{\mu} = (\boldsymbol{\mu} - \boldsymbol{\mu}_q)$. This lets us summarize the Hessian matrix $B(\boldsymbol{\mu})$:

$$B(\boldsymbol{\mu}) \;=\; -\left[K^{-1}(\Sigma_p - \Delta\boldsymbol{\mu}\,\Delta\boldsymbol{\mu}^\top)K^{-1} - K^{-1}\right] \tag{D.1}$$

This gives us the Hessian.

Hessian inversion formula. For the second order gradient descent scheme, we need to compute the inverse matrix of the Hessian $H(\boldsymbol{\beta})$. Let us call $D = \lambda_0 I$ the (diagonal) Hessian of the regularizer. We use the Woodbury matrix identity,

$$\begin{aligned} H^{-1} &= (D + X^\top B X)^{-1} \\ &= D^{-1} - D^{-1} X^\top (B^{-1} + X D^{-1} X^\top)^{-1} X D^{-1} \\ &= \lambda_0^{-1} I \lambda_0^{-2} X^\top (B^{-1} + \lambda_0^{-1} X X^\top)^{-1} X. \end{aligned} \tag{D.2}$$

Note that this identity does not require us to invert a $d \times d$ matrix, but only involves the inversion of $n \times n$ matrices. In genetic applications this form is often advantageous since the number of samples n is often smaller than the genetic features d. Furthermore, to speed up computation, we first precompute the linear kernel $XX^\top$. We also use the fact that we can more efficiently compute the product $H^{-1}\nabla_\beta \mathcal{L}$ as opposed to first calculating the inverse Hessian and then multiplying it with the gradient.

Appendix E

Generalized dynamic topic models

E.1 Derivation of the variational objective

Recall the variational objective

$$\mathcal{L}(\lambda, \phi, \mu, \Sigma) = \mathbb{E}_q[\log \tilde{p}(w|u, z)p(z|\theta)p(\theta)p(u)] - \mathbb{E}_q[\log q(\theta)q(z)q(u)],$$

where $\tilde{p}(w|u, z)$ is defined in Eq. 7.5. The first term is

$$\begin{aligned}
&\mathbb{E}_q\left[\log \tilde{p}(w|z, u)\right] \\
&= \sum_{t,n,k} \mathbb{E}_q\left[z_{tnk} \log \tilde{p}(w_{tn}|z_{tn} = k, u)\right] \\
&= \sum_{t,n,k} \mathbb{E}_q[z_{tnk}]\Big\{ \boldsymbol{K}_{t\hat{T}} K^{-1}_{\hat{T}\hat{T}} \mathbb{E}_q[u_{k..}] w_{tn} - \zeta_{kt}^{-1} \sum_v \mathbb{E}_q\left[\exp\left(\boldsymbol{K}_{t\hat{T}} K^{-1}_{\hat{T}\hat{T}} \boldsymbol{u}_{kv.} + \frac{\tilde{K}_{tt}}{2}\right)\right] \\
&\quad - \log(\zeta_{kt}) + 1\Big\} \\
&= \sum_{t,n,k} \phi_{tnk}\Big\{ \boldsymbol{K}_{t\hat{T}} K^{-1}_{\hat{T}\hat{T}} \mu_{k..} w_{tn} \\
&\quad - \zeta_{kt}^{-1} \sum_v \exp\left(\boldsymbol{K}_{t\hat{T}} K^{-1}_{\hat{T}\hat{T}} \boldsymbol{\mu}_{kv} + \frac{1}{2}(\boldsymbol{K}_{t\hat{T}} K^{-1}_{\hat{T}\hat{T}} \Sigma_{kv} K^{-1}_{\hat{T}\hat{T}} \boldsymbol{K}_{\hat{T}t} + \tilde{K}_{tt})\right) - \log(\zeta_{kt}) + 1\Big\} \\
&= \sum_{t,n,k} \phi_{tnk}\Big\{ \boldsymbol{w}_{tn}^\top \boldsymbol{m}_{k.t} - \zeta_{kt}^{-1} \sum_v \exp\left(m_{kvt} + \frac{1}{2}(\Lambda_{kvt} + \tilde{K}_{tt})\right) - \log(\zeta_{kt}) + 1\Big\},
\end{aligned}$$

where $m_{kvt} = \boldsymbol{K}_{t\hat{T}} K^{-1}_{\hat{T}\hat{T}} \boldsymbol{\mu}_{kv}$ and $\Lambda_{kvt} = \boldsymbol{K}_{t\hat{T}} K^{-1}_{\hat{T}\hat{T}} \Sigma_{kv} K^{-1}_{\hat{T}\hat{T}} \boldsymbol{K}_{\hat{T}t}$. The second term is

$$\begin{aligned}
\mathbb{E}_q[\log p(z|\theta)] &= \sum_{t,n} \mathbb{E}_q[\log p(z_{tn}|\boldsymbol{\theta}_t)] \\
&= \sum_{t,n,k} \phi_{tnk} \mathbb{E}_q[\log \theta_{tk}] \\
&= \sum_{t,n,k} \phi_{tnk} \left(\psi(\lambda_{tk}) - \psi\left(\lambda_{t0}\right)\right),
\end{aligned}$$

where $\lambda_{t0} = \sum_k \lambda_{dk}$. The negative KL terms are

$$\begin{aligned}\mathbb{E}_q[\log p(u) - \log q(u)] &= -\sum_{k,v} \mathrm{KL}(q(\boldsymbol{u}_{kv})||p(\boldsymbol{u}_{kv})) \\ &\stackrel{\mathsf{c}}{=} -\frac{1}{2}\sum_{k,v}\left(\boldsymbol{\mu}_{kv} K_{\hat{T}\hat{T}}^{-1}\boldsymbol{\mu}_{kv} + \mathrm{tr}(\Sigma_{kv}K_{\hat{T}\hat{T}}^{-1}) - \log|\Sigma_{kv}|\right),\end{aligned}$$

and

$$\begin{aligned}\mathbb{E}_q[\log p(\theta) - \log q(\theta)] &= -\sum_{t} \mathrm{KL}(q(\boldsymbol{\theta}_t)||p(\boldsymbol{\theta}_t)) \\ &\stackrel{\mathsf{c}}{=} \sum_{t,k}((\alpha_k - \lambda_{tk})(\psi(\lambda_{tk}) - \psi(\lambda_{t0})) + \log\Gamma(\lambda_{tk})) - \Gamma(\lambda_{t0}).\end{aligned}$$

The entropy of $q(z)$ is

$$-\mathbb{E}_q[q(z)] = -\sum_{t,n,k}\phi_{tnk}\log\phi_{tnk}.$$

Finally, summing all terms gives the variational objective

$$\begin{aligned}\mathcal{L}(\lambda,\phi,\mu,\Sigma) \stackrel{\mathsf{c}}{=} \sum_{t,n,k}\phi_{dnk}\Big\{\boldsymbol{w}_{tn}^\top\boldsymbol{m}_{k.t} - \zeta_{kt}^{-1}\sum_v \exp\left(m_{kvt} + \frac{1}{2}(\Lambda_{kvt} + \tilde{K}_{tt})\right) - \log(\zeta_{kt}) \\ + 1 + \psi(\lambda_{tk}) - \psi(\lambda_{t0}) - \log\phi_{tnk}\Big\} - \frac{1}{2}\sum_{k,v}\Big(\boldsymbol{\mu}_{kv}K_{\hat{T}\hat{T}}^{-1}\boldsymbol{\mu}_{kv} + \mathrm{tr}(\Sigma_{kv}K_{\hat{T}\hat{T}}^{-1}) \\ - \log|\Sigma_{kv}|\Big) + \sum_{t,k}((\alpha_k - \lambda_{tk})(\psi(\lambda_{tk}) - \psi(\lambda_{t0})) + \log\Gamma(\lambda_{tk})) - \Gamma(\lambda_{t0})\end{aligned}$$

E.2 SVI updates

In the following we provide more details on how the the parameter updates are derived.

Updating the local variables. As in standard LDA, the coordinate ascent update of $\boldsymbol{\lambda}_d$ is the expected natural parameter of the corresponding complete conditional distribution (Blei, Ng, and Jordan, 2003),

$$\boldsymbol{\lambda}_d = \alpha + \textstyle\sum_n \boldsymbol{\phi}_{tn}.$$

The derivative of $\mathcal{L}$ w.r.t. ϕ_{tnk} is

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\phi_{tnk}} = \boldsymbol{w}_{tn}^\top\boldsymbol{m}_{k.t} - \zeta_{kt}^{-1}\sum_v \exp\left(m_{kvt} + \frac{1}{2}(\Lambda_{kvt} + \tilde{K}_{tt})\right) - \log(\zeta_{kt}) + \psi(\lambda_{tk}) \\ - \psi(\lambda_{t0}) - \log\phi_{tnk}.\end{aligned}$$

Setting the derivative zero leads to

$$\phi_{tnk} = \exp\left\{\boldsymbol{w}_{tn}^{\top}\boldsymbol{m}_{k.t} - \zeta_{kt}^{-1}\sum_{v}\exp\left(m_{kvt} + \frac{1}{2}(\Lambda_{kvt} + \tilde{K}_{tt})\right) - \log(\zeta_{kt}) + \psi(\lambda_{tk}) - \psi\left(\lambda_{t0}\right)\right\}.$$

Inserting the update of the previous update of ζ_{kt} this simplifies to

$$\begin{aligned}\phi_{tnk} &= \exp\left\{\boldsymbol{w}_{tn}^{\top}\boldsymbol{m}_{k.t} - 1 - \log(\zeta_{kt}) + \psi(\lambda_{tk}) - \psi\left(\lambda_{t0}\right)\right\} \\ &\propto \exp\left\{\boldsymbol{w}_{tn}^{\top}\boldsymbol{m}_{k.t} - \log(\zeta_{kt}) + \psi(\lambda_{tk}) - \psi\left(\lambda_{t0}\right)\right\}.\end{aligned}$$

The update for the parameter vector $\phi_{tn.}$ is obtained by renormalizing (such that $||\phi_{tn.}||_1 = 1$).

Updating the global variables The standard Euclidean gradient of $\mathcal{L}$ with respect to the mean and covariance parameters of $q(u_{kv})$ is

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\boldsymbol{\mu}_{kv}} &= \boldsymbol{\Xi}_{kv} - \boldsymbol{B}_{kv} - K_{\hat{T}\hat{T}}^{-1}\boldsymbol{\mu}_{kv}, \\ \frac{\partial\mathcal{L}}{\partial\Sigma_{kv}} &= -\frac{1}{2}C_{kv} + \frac{1}{2}\Sigma_{kv}^{-1} - \frac{1}{2}K_{\hat{T}\hat{T}}^{-1},\end{aligned}$$

where

$$\begin{aligned}\boldsymbol{\Xi}_{kv} &= \sum_{t,n}\phi_{tnk}w_{tnv}K_{\hat{T}\hat{T}}^{-1}\boldsymbol{K}_{\hat{T}t} \\ \boldsymbol{B}_{kv} &= \sum_{t,n}\zeta_{kt}^{-1}\phi_{tnk}\exp\left(m_{kvt} + \frac{\Lambda_{kvt} + \tilde{K}_{tt}}{2}\right)K_{\hat{T}\hat{T}}^{-1}\boldsymbol{K}_{\hat{T}t} \\ C_{kv} &= \sum_{t,n}\zeta_{kt}^{-1}\phi_{tnk}\exp\left(m_{kvt} + \frac{\Lambda_{kvt} + \tilde{K}_{tt}}{2}\right)K_{\hat{T}\hat{T}}^{-1}\boldsymbol{K}_{\hat{T}t}\boldsymbol{K}_{t\hat{T}}K_{\hat{T}\hat{T}}^{-1}.\end{aligned}$$

We now consider the Gaussian distributions $q(u_{kv})$ in natural parametrization using $\boldsymbol{\eta}_{kv}^{(1)} = S_{kv}^{-1}\boldsymbol{\mu}_{kv}$ and $\eta_{kv}^{(2)} = -\frac{1}{2}S_{kv}^{-1}$. Applying Eq. 7.6, we obtain the natural gradient w.r.t. natural parameters,

$$\begin{aligned}\hat{\nabla}_{\boldsymbol{\eta}_{kv}^{(1)}}\mathcal{L} &= \boldsymbol{\Xi}_{kv} - \boldsymbol{B}_{kv} - K_{\hat{T}\hat{T}}^{-1}\boldsymbol{\mu}_{kv} - 2(-\frac{1}{2}C_{kv} + \frac{1}{2}\Sigma_{kv}^{-1} - \frac{1}{2}K_{\hat{T}\hat{T}}^{-1})\boldsymbol{\mu}_{kv} \\ &= \boldsymbol{\Xi}_{kv} + \boldsymbol{B}_{kv}\circ(\boldsymbol{m}_{kv} - 1) - \boldsymbol{\eta}_{kv}^{(1)}\end{aligned}$$

and

$$\hat{\nabla}_{\eta_{kv}^{(2)}} \mathcal{L} = -\frac{1}{2} C_{kv} - \frac{1}{2} K_{\hat{T}\hat{T}}^{-1} - \eta_{kv}^{(2)}.$$

Note that $\boldsymbol{m}_{kv}$ as a function of the natural parameters is

$$\boldsymbol{m}_{kv} = \boldsymbol{K}_{t\hat{T}} K_{\hat{T}\hat{T}}^{-1} \boldsymbol{\mu}_{kv} = -\frac{1}{2} \boldsymbol{K}_{t\hat{T}} K_{\hat{T}\hat{T}}^{-1} \left(\eta_{kv}^{(2)}\right)^{-1} \boldsymbol{\eta}_{kv}^{(1)}.$$

E.3 Global td-idf score

To determine important words, we use an extension to the classic tf-idf scoring scheme. The score of a word is

$$\text{score}(w) = \frac{n_w}{M} \log\left(\frac{D}{n_{dw}}\right),$$

where M is the total amount of terms in the corpus, D is the number of documents, n_{dw} is the frequency of word w in document d and $n_w = \sum_d n_{dw}$.

Bibliography

Agovic, Amrudin and Arindam Banerjee (2010). "Gaussian Process Topic Models". In: *Conference on Uncertainty in Artificial Intelligence*.

Albert, James H. and Siddhartha Chib (1993). "Bayesian analysis of binary and polychotomous response data". In: *Journal of the American Statistical Association* 88.422, pp. 669–679.

Amari, Shun-ichi and Hiroshi Nagaoka (2007). *Methods of Information Geometry*. American Mathematical Society.

Arp, Daniel et al. (2014). "DREBIN: Effective and Explainable Detection of Android Malware in Your Pocket". In: *Proceedings of NDSS*.

Arthur, David and Sergei Vassilvitskii (2007). "k-means++: The advantages of careful seeding". In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035.

Astle, William and David J Balding (2009). "Population Structure and Cryptic Relatedness in Genetic Association Studies". In: *Statistical Science*, pp. 451–471.

Atwell, Susanna et al. (2010). "Genome-wide association study of 107 phenotypes in Arabidopsis thaliana inbred lines". In: *Nature* 465.7298, pp. 627–631.

Bachem, Olivier et al. (2016). "Fast and Provably Good Seedings for k-Means". In: *Neural Information Processing Systems*.

Baldi, P, P Sadowski, and D Whiteson (2014). "Searching for exotic particles in high-energy physics with deep learning." In: *Nature communications*.

Bamler, Robert and Stephan Mandt (2017). "Dynamic Word Embeddings via Skip-Gram Filtering". In: *arXiv preprint arXiv:1702.08359*.

Beckers, T., D. Kulić, and S. Hirche (2019). "Stable Gaussian Process based Tracking Control of Euler-Lagrange Systems". In: *Automatica* 103, pp. 390–397.

Berry, Matthew et al. (2010). "An interferon-inducible neutrophil-driven blood transcriptional signature in human tuberculosis". In: *Nature* 466.7309, pp. 973–977.

Bhadury, Arnab et al. (2016). "Scaling up dynamic topic models". In: *Proceedings of WWW*.

Bishop, Christopher M (2006). *Pattern recognition and machine learning*. Information Science and Statistics. Springer, New York.

Blei, David M (2012). "Probabilistic topic models". In: *Communications of the ACM* 55.4, pp. 77–84.

Blei, David M., Alp Kucukelbir, and Jon D. McAuliffe (2017). "Variational Inference: A Review for Statisticians". In: *Journal of the American Statistical Association*.

Blei, David M and John D Lafferty (2006). "Dynamic topic models". In: *Proceedings of ICML*.
— (2007). "A Correlated Topic Model of Science". In: *The Annals of Applied Statistics*.
— (2009). "Visualizing topics with multi-word expressions". In: *arXiv:0907.1013*.
Blei, David M, Andrew Y Ng, and Michael I Jordan (2003). "Latent dirichlet allocation". In: *The Journal of Machine Learning Research* 3, pp. 993–1022.
Bojarski, Mariusz et al. (2016). "End to End Learning for Self-Driving Cars." In: *CoRR* abs/1604.07316.
Boyd, Stephen et al. (2011). "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers". In: *Foundations and Trends in Machine Learning* 3.1, pp. 1–122.
Brier, Glenn W (1950). "Verification of forecasts expressed in terms of probability". In: *Monthly weather review* 1, pp. 1–3.
Buchholz, Alexander, Florian Wenzel, and Stephan Mandt (2018). "Quasi-Monte Carlo Variational Inference". In: *International Conference on Machine Learning*, pp. 667–676.
Carbonetto, Peter, Matthew Stephens, et al. (2012). "Scalable variational inference for Bayesian variable selection in regression, and its accuracy in genetic association studies". In: *Bayesian analysis* 7.1, pp. 73–108.
Caruana, Rich et al. (2015). "Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-day Readmission". In: *KDD*. ACM, pp. 1721–1730.
Cesnovar, Rok and Erik Strumbelj (2017). "Bayesian Lasso and multinomial logistic regression on GPU". In: *PLOS ONE* 12.6, pp. 1–17.
Chai, Kian Ming Adam (2012). "Variational Multinomial Logit Gaussian Process". In: *Journal of Machine Learning Research* 13, pp. 1745–1808.
Chang, Chih-Chung and Chih-Jen Lin (2011). "LIBSVM: A library for support vector machines". In: *ACM Transactions on Intelligent Systems and Technology* (3), 27:1–27:27.
Chang, Chih-chung and Chih-jen Lin (2013). "LIBSVM : A Library for Support Vector Machines". In: *ACM Transactions on Intelligent Systems and Technology (TIST)*, pp. 1–39.
Charlin, Laurent and Richard S Zemel (2013). "The Toronto paper matching system: an automated paper-reviewer assignment system". In: *Proceedings of ICML*.
Chong, Wang, David Blei, and Fei-Fei Li (2009). "Simultaneous image classification and annotation". In: *Proceedings of CVPR*.
Cortes, Corinna and Vladimir Vapnik (1995). "Support-Vector Networks". In: *Machine Learning*.
Craddock, Nick, Matthew E Hurles, Niall Cardin, et al. (2010). "Genome-wide association study of CNVs in 16,000 cases of eight common diseases and 3,000 shared controls". In: *Nature* 464.7289, pp. 713–720.
Csató, Lehel (2002). "Gaussian Processes - Iterative Sparse Approximations". PhD thesis.
Csató, Lehel and Manfred Opper (2002). "Sparse On-line Gaussian Processes". In: *Neural Comput.* 14.3, pp. 641–668.

Cunningham, John P, Philipp Hennig, and Simon Lacoste-Julien (2011). "Gaussian Probabilities and Expectation Propagation". In: *arXiv preprint: arXiv:1111.6832*.

Damianou, Andreas and Neil Lawrence (2013). "Deep Gaussian Processes". In: *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pp. 207–215.

De G. Matthews, Alexander G. et al. (2017). "GPflow: A Gaussian Process Library Using Tensorflow". In: *JMLR* 18.1, pp. 1299–1304.

Dempster, Arthur P, Nan M Laird, and Donald B Rubin (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm". In: *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38.

Dezfouli, Amir and Edwin V. Bonilla (2015). "Scalable Inference for Gaussian Process Models with Black-Box Likelihoods". In: *Advances in Neural Information Processing Systems*, pp. 1414–1422.

Diethe, T. (2015). *13 benchmark datasets derived from the UCI, DELVE and STATLOG repositories*.

Donner, Christian and Manfred Opper (2017). "The inverse Ising problem in continuous time: A latent variable approach". In: *Physical Review E* 96.6, p. 062104.

— (2018). "Efficient Bayesian Inference for a Gaussian Process Density Model". In: *Proceedings of UAI*, pp. 1–10.

Duvenaud, David (2014). "Automatic model construction with Gaussian processes". PhD thesis.

Eckstein, Jonathan and Dimitri P. Bertsekas (1992). "On the Douglas-Rachford Splitting Method and the Proximal Point Algorithm for Maximal Monotone Operators". In: *Math. Program.* 55.3, pp. 293–318.

Eleftheriadis, S. et al. (2017). "Gaussian Process Domain Experts for Modeling of Facial Affect". In: *IEEE Transactions on Image Processing* 26.10, pp. 4697–4711.

Fawcett, Tom (2006). "An introduction to ROC analysis". In: *Pattern recognition letters* 27.8, pp. 861–874.

Fei-Fei, Li and Pietro Perona (2005). "A bayesian hierarchical model for learning natural scene categories". In: *Proceedings of CVPR*.

Fernández-Delgado, Manuel et al. (2014). "Do We Need Hundreds of Classifiers to Solve Real World Classification Problems?" In: *JMLR*.

Fletcher, Roger (1987). *Practical Methods of Optimization*. Wiley-Interscience New York.

Fusi, Nicoló, Oliver Stegle, and Neil D Lawrence (2012). "Joint Modelling of Confounding Factors and Prominent Genetic Regulators Provides Increased Accuracy in Genetical Studies". In: *PLoS comp. bio.* 8.1.

G. Matthews, Alexander G. de et al. (2018). "Gaussian Process Behaviour in Wide Deep Neural Networks". In: *arXiv*.

Garnelo, Marta et al. (2018). "Neural Processes". In: *arXiv*.

Ghahramani, Z (2015). "Probabilistic machine learning and artificial intelligence". In: *Nature* 521, pp. 452–459.

Gibbs, M. N. and D. J. C. MacKay (2000). "Variational Gaussian Process Classifiers". In: *IEEE Transactions on Neural Networks* 11.6, pp. 1458–1464.

Girolami, Mark and Simon Rogers (2006). "Variational Bayesian multinomial probit regression with Gaussian process priors". In: *Neural Computation* 18.8, pp. 1790–1817.

Gopalan, Prem et al. (2016). "Scaling probabilistic models of genetic variation to millions of humans". In: *Nature Genetics* 48.12.

Guo, Chuan et al. (2017). "On Calibration of Modern Neural Networks". In: *ICML*. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 1321–1330.

Henao, Ricardo, Xin Yuan, and Lawrence Carin (2014). "Bayesian Nonlinear Support Vector Machines and Discriminative Factor Modeling". In: *Neural Information Processing Systems*.

Hennig, Philipp et al. (2012). "Kernel Topic Models". In: *Proceedings of AISTATS*.

Hensman, James, Nicolo Fusi, and Neil D Lawrence (2013). "Gaussian processes for big data". In: *Conference on Uncertainty in Artificial Intellegence*.

Hensman, James and Alexander Matthews (2015). "Scalable Variational Gaussian Process Classification". In: *International Conference on Artificial Intelligence and Statistics*.

Hensman, James et al. (2015). "MCMC for Variationally Sparse Gaussian Processes". In: *Neural Information Processing Systems*.

Hernández-Lobato, Daniel, José M Hernández-Lobato, and Pierre Dupont (2011). "Robust multi-class Gaussian process classification". In: *Advances in neural information processing systems*, pp. 280–288.

Hernández-Lobato, Daniel and José Miguel Hernández-Lobato (2016). "Scalable Gaussian Process Classification via Expectation Propagation". In: *AISTATS*, pp. 168–176.

Herzog, Stefan and Dirk Ostwald (2013). "Sometimes Bayesian statistics are better". In: *Nature*.

Hoffman, Matthew D. et al. (2013). "Stochastic Variational Inference". In: *JMLR*.

Imbens, Guido W and Donald B Rubin (2015). *Causal Inference in Statistics, Social, and Biomedical Sciences*. Cambridge University Press.

Izmailov, Pavel, Alexander Novikov, and Dmitry Kropotov (2018). "Scalable Gaussian Processes with Billions of Inducing Inputs via Tensor Train Decomposition". In: *International Conference on Artificial Intelligence and Statistics*.

Jacot, Arthur, Franck Gabriel, and Clément Hongler (2018). "Neural Tangent Kernel: Convergence and Generalization in Neural Networks". In: *International Conference on Neural Information Processing Systems*.

John Walker, Saint (2014). *Big data: A revolution that will transform how we live, work, and think*. Taylor & Francis.

Jørgensen, Bent (2012). *Statistical properties of the generalized inverse Gaussian distribution*. Springer Science & Business Media.

Khan, Mohammad Emtiyaz and Didrik Nielsen (2018). "Fast yet Simple Natural-Gradient Descent for Variational Inference in Complex Models". In: *Arxiv Preprint*.

Kim, Hyun-Chul and Zoubin Ghahramani (2006). "Bayesian Gaussian Process Classification with the EM-EP Algorithm". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 28.12, pp. 1948–1959.

Kingma, Diederik P and Jimmy Ba (2015). "Adam: A method for stochastic optimization". In: *Proceedings of the 3rd International Conference on Learning Representations*.

Klasen, Jonas R et al. (2016). "A multi-marker association method for genome-wide association studies without the need for population structure correction". In: *Nature communications* 7, p. 13299.

Kloft, M et al. (2011). "Lp-norm multiple kernel learning". In: *The Journal of Machine Learning Research* 12, pp. 953–997.

Kraft, Peter, Eleftheria Zeggini, and John PA Ioannidis (2009). "Replication in genome-wide association studies". In: *Statistical Science: A review journal of the Institute of Mathematical Statistics* 24.4, p. 561.

Krzywinski, Martin and Naomi Altman (2013). "Importance of being uncertain". In: *Nature Methods*.

Lawrence, Neil D., Magnus Rattray, and Michalis K. Titsias (2009). "Efficient Sampling for Gaussian Process Inference using Control Variables". In: *Advances in Neural Information Processing Systems 21*, pp. 1681–1688.

Li, Limin, Barbara Rakitsch, and Karsten M. Borgwardt (2011). "ccSVM: correcting Support Vector Machines for confounding factors in biological data classification". In: *Bioinformatics* 27.13, pp. 342–348.

Lichman, M. (2013). *UCI Machine Learning Repository*.

Linderman, Scott W., Matthew J. Johnson, and Ryan P. Adams (2015). "Dependent Multinomial Models Made Easy: Stick-Breaking with the Polya-gamma Augmentation". In: *Neural Information Processing Systems*.

Lippert, C et al. (2011). "FaST linear mixed models for genome-wide association studies". In: *Nature Methods* 8.10, pp. 833–835.

Lippert, Christoph (2013). "Linear mixed models for genome-wide association studies". PhD thesis. Eberhard Karls Universität Tübingen.

Luts, Jan and John T. Ormerod (2014). "Mean Field Variational Bayesian Inference for Support Vector Machine Classification". In: *Comput. Stat. Data Anal.* Pp. 163–176.

Ma, Chao, Yingzhen Li, and Jose Miguel Hernandez-Lobato (2019). "Variational Implicit Processes". In: *International Conference on Machine Learning*.

Mandt, S., M. Hoffman, and D. Blei (2016). "A Variational Analysis of Stochastic Gradient Algorithms". In: *International Conference on Machine Learning*.

Manolio, Teri A et al. (2009). "Finding the Missing Heritability of Complex Diseases". In: *Nature* 461.7265, pp. 747–753.

Maritz, J.S. and T. Lwin (1989). "Empirical Bayes Methods with Applications". In: *Monographs on Statistics and Applied Probability*.

Matthews, Alexander G. de G. et al. (2017). "GPflow: A Gaussian process library using TensorFlow". In: *Journal of Machine Learning Research*.

McCallum, Andrew, Andrés Corrada-Emmanuel, and Xuerui Wang (2004). "The Author-Recipient-Topic Model for Topic and Role Discovery in Social Networks: Experiments with Enron and Academic Email". In: *Workshop on Structured Data and Representations in Probabilistic Models for Categorization at NIPS*.

Meinshausen, Nicolai and Peter Bühlmann (2010). "Stability Selection". In: *Journal of the Royal Statistical Society, Series B* 72, pp. 417–473.

Michelmore, Rhiannon, Marta Kwiatkowska, and Yarin Gal (2018). "Evaluating Uncertainty Quantification in End-to-End Autonomous Driving Control". In:

Minka, Thomas P (2001). "Expectation Propagation for Approximate Bayesian Inference". In: *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pp. 362–369.

Mohamed, Shakir, Katherine Heller, and Zoubin Ghahramani (2011). "Bayesian and L1 Approaches for Sparse Unsupervised Learning". In: *arXiv:1106.1157*.

Mohri, Mehryar, Afshin Rostamizadeh, and Ameet Talwalkar (2012). *Foundations of machine learning*. MIT press.

Morgan, Stephen L and Christopher Winship (2014). *Counterfactuals and Causal Inference*. Cambridge University Press.

Murphy (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press.

Murray, Iain, Ryan Prescott Adams, and David J. C. MacKay (2010). "Elliptical slice sampling". In: *The Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pp. 541–548.

Naeini, Mahdi Pakdaman, Gregory Cooper, and Milos Hauskrecht (2015). "Obtaining well calibrated probabilities using bayesian binning". In: *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Nuzzo, R. (2014). In: *Nature* 506.

Opper, Manfred and Cédric Archambeau (2009). "The Variational Gaussian Approximation Revisited". In: *Neural Comput.* 21.3, pp. 786–792.

Opper, Manfred and Ole Winther (2001). "From Naive Mean Field Theory to the TAP Equations". In: *Advanced Mean Field Methods: Theory and Practice*.

Owen, A.B. (1998). "Monte Carlo extension of quasi-Monte Carlo". In: *1998 Winter Simulation Conference. Proceedings (Cat. No.98CH36274)* 1.1, pp. 571–577.

Pandit, Ravi Kumar and David Infield (2018). "Comparative analysis of binning and Gaussian Process based blade pitch angle curve of a wind turbine for the purpose of condition monitoring". In: *Journal of Physics: Conference Series* 1102.

Park, Chiwoo and Daniel Apley (2018). "Patchwork Kriging for Large-scale Gaussian Process Regression". In: *Journal of Machine Learning Research* 19.7, pp. 1–43.

Pearl, Judea et al. (2009). "Causal inference in statistics: An overview". In: *Statistics Surveys* 3, pp. 96–146.

Platt, J. C. (1999). "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods". In: *Advances in Large Margin Classifier*.

Polson, Nicholas G., James G. Scott, and Jesse Windle (2013). "Bayesian Inference for Logistic Models Using Pólya–Gamma Latent Variables". In: *Journal of the American Statistical Association* 108.504, pp. 1339–1349.

Polson, Nicholas G. and Steven L. Scott (2011). "Data augmentation for support vector machines". In: *Bayesian Anal.*

Prékopa, András (1973). "On logarithmic concave measures and functions". In: *Acta Scientiarum Mathematicarum* 34, pp. 35–343.

Price, A L et al. (2006). “Principal components analysis corrects for stratification in genome-wide association studies”. In: *Nature Genetics* 38.8, pp. 904–909.

Price, Alkes L et al. (2010). “New approaches to population stratification in genome-wide association studies”. In: *Nature Reviews Genetics* 11.7, pp. 459–463.

Pritchard, Jonathan K, Matthew Stephens, and Peter Donnelly (2000). “Inference of population structure using multilocus genotype data”. In: *Genetics* 155.2, pp. 945–959.

Rakitsch, B. et al. (2013). “A Lasso Multi-Marker Mixed Model for Association Mapping with Population Structure Correction”. In: *Bioinformatics* 29.2, pp. 206–214.

Ranganath, Rajesh et al. (2013). “An Adaptive Learning Rate for Stochastic Variational Inference”. In: *International Conference on Machine Learning*.

Rasmussen, C. E. and C. K. I. Williams (2006). *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press.

Riihimäki, Jaakko, Pasi Jylänki, and Aki Vehtari (2013). “Nested Expectation Propagation for Gaussian Process Classification”. In: *JMLR* 14.1, pp. 75–109.

Roberts, Stephen et al. (2012). “Gaussian Processes for Timeseries Modelling”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 371.

Ruiz, Francisco J. R. et al. (2018). “Augment and Reduce: Stochastic Inference for Large Categorical Distributions”. In: *International Conference on Machine Learning*.

Sæmundsson, Steindór, Katja Hofmann, and Marc Peter Deisenroth (2018). “Meta Reinforcement Learning with Latent Variable Gaussian Processes”. In: *Uncertainty in Artificial Intelligence (UAI)*.

Salimbeni, Hugh, Stefanos Eleftheriadis, and James Hensman (2018). “Natural Gradients in Practice: Non-Conjugate Variational Inference in Gaussian Process Models”. In: *International Conference on Artificial Intelligence and Statistics*.

Sandhaus, Evan (2008). *The New York Times Annotated Corpus*. Linguistic Data Consortium.

Schoenberg, Isaac J (1938). “Metric spaces and completely monotone functions”. In: *Annals of Mathematics*, pp. 811–841.

Scott, James G and Liang Sun (2013). “Expectation-maximization for logistic regression”. In: *arXiv preprint arXiv:1306.0040*.

Seeger, Matthias, Christopher K. I. Williams, and Neil D. Lawrence (2003). “Fast Forward Selection to Speed Up Sparse Gaussian Process Regression”. In: *International Conference on Artificial Intelligence and Statistics*.

Seeger, Matthias W and Hannes Nickisch (2011). “Large Scale Bayesian Inference and Experimental Design for Sparse Linear Models”. In: *SIAM Journal on Imaging Sciences* 4.1, pp. 166–199.

Shepero, Mahmoud et al. (2018). “Residential probabilistic load forecasting: A method using Gaussian process designed for electric load data”. In: *Applied Energy* 218.C, pp. 159–172.

Snelson, Edward and Zoubin Ghahramani (2005). “Sparse Gaussian processes using pseudo-inputs”. In: pp. 1257–1264.

Song, Minsun, Wei Hao, and John D Storey (2015). "Testing for genetic associations in arbitrarily structured populations". In: *Nature genetics* 47.5, pp. 550–554.

Stein, Michael L (2012). *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media.

Tibshirani, Robert (1996). "Regression Shrinkage and Selection via the Lasso". In: *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288.

Titsias, Michalis (2016). "One-vs-each approximation to softmax for scalable estimation of probabilities". In: *Advances in Neural Information Processing Systems*, pp. 4161–4169.

Titsias, Michalis K. (2009). "Variational learning of inducing variables in sparse Gaussian processes". In: *In Artificial Intelligence and Statistics 12*, pp. 567–574.

Vattikuti, Shashaank et al. (2014). "Applying compressed sensing to genome-wide association studies". In: *GigaScience* 3.1, p. 10.

Vilhjálmsson, Bjarni J and Magnus Nordborg (2013). "The nature of confounding in genome-wide association studies". In: *Nature Reviews Genetics* 14.1, pp. 1–2.

Villacampa-Calvo, Carlos and Daniel Hernández-Lobato (2017). "Scalable Multi-Class Gaussian Process Classification using Expectation Propagation". In: *International Conference on Machine Learning*.

Wainwright, Martin J and Michael I Jordan (2007). "Graphical Models, Exponential Families, and Variational Inference". In: *Foundations and Trends in Machine Learning* 1.1–2, pp. 1–305.

Wainwright, Martin J. and Michael I. Jordan (2008). "Graphical Models, Exponential Families, and Variational Inference". In: *Foundations and Trends in Machine Learning* 1-2, pp. 1–305.

Walker, Stephen G (2011). "Posterior sampling when the normalizing constant is unknown". In: *Communications in Statistics–Simulation and Computation* 40.5, pp. 784–792.

Wang, Chong, David Blei, and David Heckerman (2008). "Continuous Time Dynamic Topic Models". In: *Proceedings of UAI*.

Wang, Chong and David M Blei (2011). "Collaborative topic modeling for recommending scientific articles". In: *Proceedings of ACM SIGKDD*.

Wang, Xuerui and Andrew McCallum (2006). "Topics over time: a non-Markov continuous-time model of topical trends". In: *Proceedings of ACM SIGKDD*.

Wang, Xuerui, Andrew McCallum, and X Wei (2007). "Topical N-Grams: Phrase and Topic Discovery, with an Application to Information Retrieval". In: *Data Mining*.

Welling, Max and Yee-Whye Teh (2011). "Bayesian learning via stochastic gradient Langevin dynamics". In: *Proceedings of ICML*, pp. 681–688.

Williams, Christopher K. I. and David Barber (1998). "Bayesian Classification with Gaussian Processes". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20, pp. 1342–1351.

Wilson, Andrew Gordon et al. (2016). "Deep Kernel Learning". In: *International Conference on Artificial Intelligence and Statistics*, pp. 370–378.

Xiong, Haitao, Junjie Wu, and Lu Liu (2010). "Classification with ClassOverlapping: A Systematic Study". In: *Proceedings of the 1st International Conference on E-Business Intelligence (ICEBI2010),* Atlantis Press.

Xiong, Wayne et al. (2016). "Achieving Human Parity in Conversational Speech Recognition". In: *CoRR* abs/1610.05256.

Xu, Minjie, Jun Zhu, and Bo Zhang (2013). "Fast Max-Margin Matrix Factorization with Data Augmentation". In: *International Conference on Machine Learning*, pp. 978–986.

Zhang, Jun, and Zhang (2014). "Max-Margin Infinite Hidden Markov Models". In: *International Conference on Machine Learning*.

Zhu, Jun et al. (2014). "Gibbs Max-margin Topic Models with Data Augmentation". In: *JMLR*.

Declaration of independent work (Selbständigkeitserklärung)

Ich erkläre, dass ich die Dissertation selbständig und nur unter Verwendung der von mir gemäß 7 Abs. 3 der Promotionsordnung der Mathematisch-Naturwissenschaftlichen Fakultät, veröffentlicht im Amtlichen Mitteilungsblatt der Humboldt-Universität zu Berlin Nr. 42/2018 am 11.07.2018 angegebenen Hilfsmittel angefertigt habe.

Florian Wenzel
September 2019